2

7

HUGHES

HUGHES AIRCRAFT COMPANY
GROUND SYSTEMS GROUP

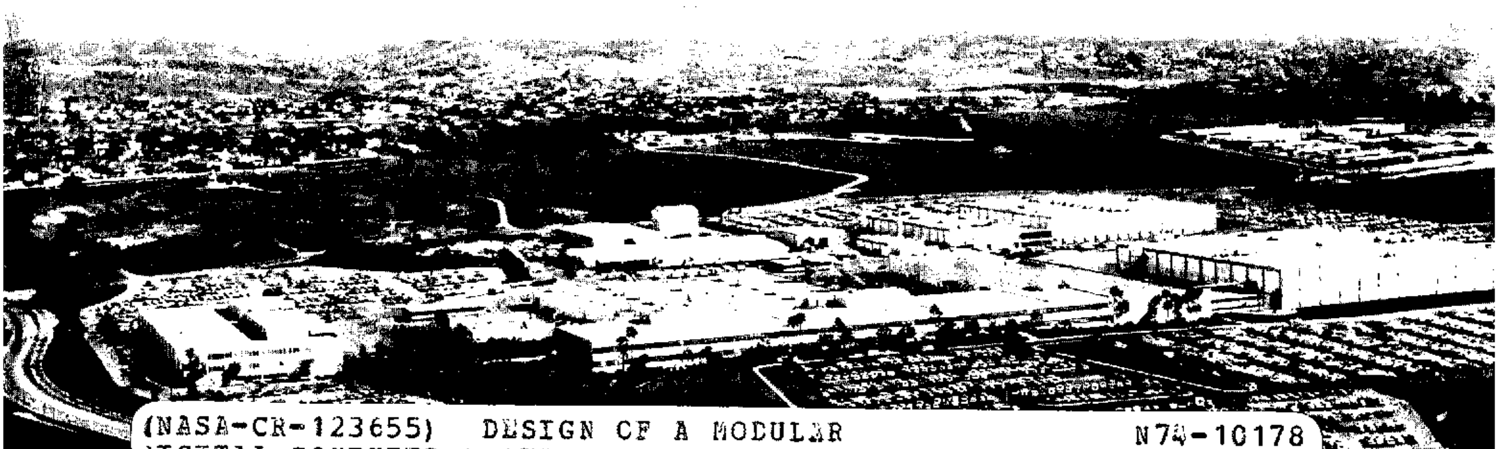
CR-123555

MAY 1972
RECEIVED
NASA STI FACILITY
INPUT BRANCH

DESIGN OF
A MODULAR DIGITAL COMPUTER SYSTEM
DRL 4 PHASE I REPORT (U)

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
U S Department of Commerce
Springfield, VA. 22151

CONTRACT NO. NAS8-27926



(NASA-CR-123655) DESIGN OF A MODULAR DIGITAL COMPUTER SYSTEM, DRL 4 Phase 1 Report, 8 Oct. 1971 - 15 Mar. 1972 (Hughes Aircraft Co.)

N74-10178

CSCL 09B G3/08

Unclas
20811

DESIGN OF A MODULAR DIGITAL COMPUTER SYSTEM

DRL 4

PHASE I REPORT

April 15, 1972

Prepared under Contract NAS8-27926

by

HUGHES AIRCRAFT COMPANY
FULLERTON, CALIF.

for

George C. Marshall Space Flight Center
National Aeronautics and Space Administration

FR 72-11-450

FOREWORD

This report documents the accomplishments of Phase I of contract NAS8-27926 whose scope is the design of an Automatically Reconfigurable Modular Multiprocessor System (ARMMS). The Phase I time period was from October 8, 1971 to March 15, 1972. The design is being performed by the Data Processing Products Division of Hughes-Fullerton. M & S Computing, Inc. is providing support in the area of executive software design under subcontract to Hughes. The design is being directed by the Astrionics Laboratory of the Marshall Space Flight Center. The Contracting Officer's Representative is Dr. J. B. White.

In accordance with the data requirements of NAS8-27926, this report consists primarily of reproductions of internal reports which have not been edited nor retyped for this report. As such, it reflects the evolution of the design rather than its culmination and must be read from that perspective. Each report is preceded by a brief discussion of its content and conclusions.

Major individual contributors to the report include the following:

W. L. Martin	ARMMS Phase I Architecture
J. H. Engleman	Preliminary ARMMS Reliability Feasibility Study ARMMS Reliability Data Base
W. G. Tees	Partitioning Study of the SUMC Processor
J. L. Bricker	A Unified Method for Analyzing Mission Profile Reliability
B. Cohen S. A. Simpson	Investigation of Inter-Module Data Transmission Rates for ARMMS
T. T. Schansman (M&S) E. I. Eastin (M&S) K. H. Schonrock (M&S)	Mission Analysis Profile Synchronous vs. Non-Synchronous Scheduling Control

CONTENTS

SECTION 1

SUMMARY OF PHASE I OF THE ARMMS DESIGN

The primary objective of contract NAS8-27926 is to perform the system design of an advanced modular computer system designated the Automatically Reconfigurable Modular Multiprocessor System (ARMMS). ARMMS is addressing the anticipated requirements for both higher computing capacity and reliability which may characterize spaceborne computers in the late 1970's to mid-1980's. ARMMS is intended to achieve both of these objectives through a highly modular computer architecture which can be configured as a multiprocessor for maximum computing speed or as a triple modular redundant (TMR) system with standby spares for extremely high reliability. Moreover, the configuration will be dynamic; that is, it will be possible to change the configuration in real time as needed by various mission phases or events. A peak computing capacity of several million instructions per second is planned, while the probability of a minimum computing capacity surviving for 5 years is to be 0.99.

ARMMS is an outgrowth and extension of two NASA development programs, the MSFC Space Ultrareliable Modular Computer (SUMC) and the ERC Modular Computer. The SUMC program has emphasized the development of a processor which is effectively partitioned for LSI implementation. To date, a breadboard TTL prototype has been constructed and a MOS LSI version is nearing completion.

A version of SUMC is anticipated to be the processor module of the ARMMS system. The breadboard of the ERC Modular Computer is undergoing evaluation at MSFC, and the experience gained will be relevant to ARMMS. The ERC Modular Computer had the common objective with ARMMS of achieving a variable configuration for varying levels of processing capacity and reliability.

In addition, the experience of numerous NASA, Air Force, and Navy architectur and design studies is being reviewed and incorporated into the ARMMS design wher appropriate. In general, these efforts have considered a subset of the ARMMS objectives. For example, the JPL STAR is oriented toward long-life reliability. The MSC reconfigurable guidance and control computer study considers primarily space shuttle requirements. Other studies have considered space station compute requirements. All have identified design principles which form a substantial base of experience for the ARMMS development.

The 18-month contract is divided into three phases. The overall plan is shown in Figure 1. At the inception of the contract, an initial baseline description was provided by MSFC. The primary efforts in Phase I have been to establish general design guidelines necessary to achieve the ARMMS reliability and performance objectives; to survey published estimates of performance requirements for future space computers, and to refine the initial baseline.

The specific objectives to be achieved within the 18 month period are the following:

1. To perform the detailed design (to the gate level) of all module interfaces and switches.
2. To define the design of all ARMM modules types to the detailed block-diagram (register) level.
3. To perform the functional design of the executive software as it pertains to error detection and correction. (M & S Computing, Inc. is performing this task under subcontract to Hughes.)
4. To define the overall system response to all classes of failures.
5. To develop sample packaging concepts for an eventual implementation of ARMMS.
6. To simulate the computational performance of ARMMS in its high computing capacity mode.
7. To develop and apply reliability models as needed to support the design.

As specified in the contract data requirements, the Phase Reports are to consist primarily of reproductions of contractor internal documents written during that phase. The remaining sections of this report consist of documents prepared at various stages of Phase I. The general subject of each is listed below:

Section 2. ARMMS Architecture Status - Phase I

An overview of the architecture as modified and refined in Phase I is given.

Section 3. Mission Analysis Profile

The data extracted from more than 30 sources concerning mission requirements is presented.

Section 4. Synchronous vs. Non-Synchronous Scheduling Control

As a first step in the executive software design, the two major scheduling approaches are discussed.

Section 5. Preliminary ARMMS Reliability Feasibility Study

An initial model was programmed to allow design requirements for the 5-year survival of a simplex computer to be explored.

Section 6. ARMMS Reliability Data Base

Based on a survey of available data, a set of part failure rates have been tabulated for use in future analysis.

Section 7. Results of the Initial Partitioning Study of the SUMC Processor

It was initially believed that the reliability objective would necessitate extensive subpartitioning. The SUMC logic design was studied in an effort to discover reasonable approaches to subpartitioning without requiring redesign.

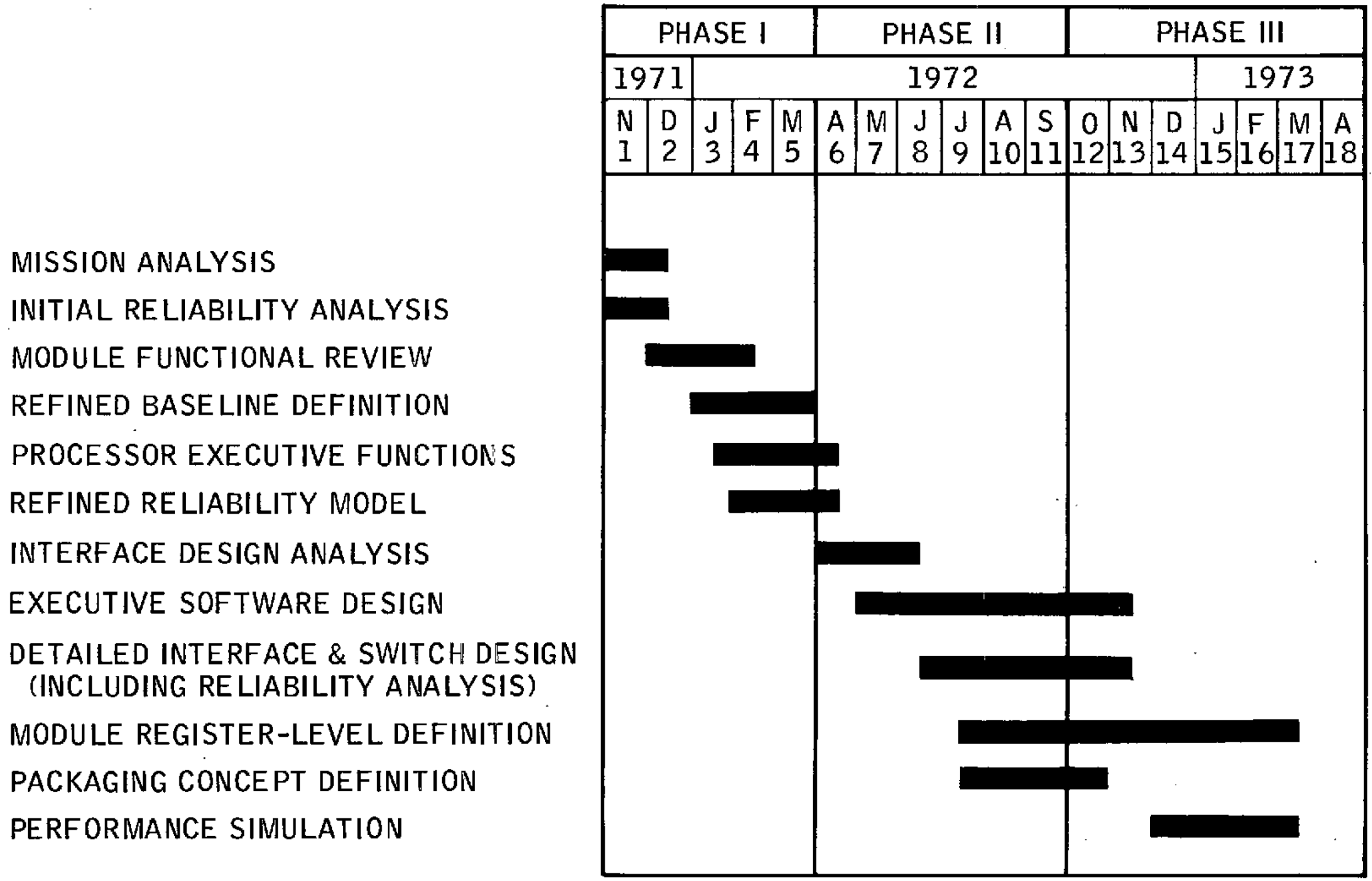
Section 8. Data Transmission Study

A key design factor ARMMS will be the characteristics of the inter-module data paths. An initial study of feasible speeds and interconnection methods is presented.

Appendix A A Unified Method For Analyzing Mission Profile Reliability For Standby and Multiple Modular Redundant Computing Systems which Allows for Degraded Performance

This appendix is the manuscript of a paper submitted to the IEEE Transactions on Reliability Theory describing the second reliability model developed for the ARMMS program.

PROGRAM PLAN



25160-55

HUGHES
HUGHES AIRCRAFT COMPANY
GROUND SYSTEMS GROUP

SECTION 2

ARMMS Architecture Status - Phase I

The report contained in this section was the first attempt to use the efforts of the mission analysis, reliability, sub-partitioning, and literature search to define a refined ARMMS baseline concept. Its contents are summarized in the first section.

ARMMS ARCHITECTURE STATUS - PHASE 1

I. Introduction and Summary

This report has the purpose of synthesizing the results of all ongoing ARMMS design efforts as they have impacted and extended the ARMMS baseline architecture as of Phase I of the contract. It is the first refinement of the Baseline as defined at the inception of the contract and as such makes no presumption of being definitive. Similar re-definitions may be expected at least at the end of each phase of the program. For convenience, the original baseline description is character-ized as "Baseline 0" and the description provided here is called "Baseline 1". Baseline 1 is perhaps more specific in the aspects of the design discussed although it does not yet encompass the broad scope of topics (e.g. power supply redundancy) touched upon by Baseline 0.

The specific topics discussed in the body of this report are listed below:

1. The possible major submodes of operation of ARMMS are identified along with possible limitations. It is specifically recommended that no more than one TMR processor set at a time should be required and that no more than three parallel instruction streams should be required. It is also recommended that the inclusion of a duplex processor mode be considered as the design proceeds.

2. The question of whether a dedicated executive should be retained or whether a floating executive concept should be reconsidered is discussed. The conclusion is that the dedicated executive module should be retained in the baseline concept.

3. The relative merits of task memory, cache memory, and local storage are discussed. It is concluded that small local storage (e.g. 128 words) in each processor module is preferable to the larger (e.g. 4 K words) task memory modules.

4. A first attempt at detailing the data and control paths of the ARMMS system is presented, including estimated pin counts and interface speed requirements. Overall summaries of module pin counts and number of voters are given. Although the analysis given is particularly tenuous, it still provides a framework for discussion and for proceeding to the next level of design detail.

II. Restatement of the Fundamental Design Objectives of ARMMS

Any computer system justifies the cost of its development to the degree that it provides new capabilities or allows earlier ones to be satisfied at reduced cost. ARMMS is primarily oriented toward providing the following new capabilities for spaceborne computers for application in the 1975 to 1985 time period.

1. To provide a modular computer system which is responsive to many mission types and phases.

2. To achieve through modularity a higher computing capability than previously available for spaceborne application. A target of several million instructions per second has been chosen.

3. To provide the capability to choose to maximize reliability through the use of redundancy or to maximize processing capacity through multiprocessing. Moreover, this multi-mode capability must be dynamic; that is, a given system may alternate from one mode to another as a function of real-time requirements.

4. To maximize reliability in all applications through the incorporation of fault detection and recovery features and through the use of high reliability components.

The first consideration of any ARMMS design tradeoff is to avoid compromising these basic objectives. However, an advanced paper design will surely remain only that unless continuous concern is is maintained for the practical requirements of implementation. Such design parameters as power density, weight, volume, pin count, device count, etc., must influence the design process.

The evolving baseline as presented here is oriented toward achieving the ARMMS objectives within a practical hardware and software context.

III. Major Phase 1 Factors Affecting Baseline I

Baseline I as described in this document, although not different in the objectives it addresses from Baseline 0, is somewhat different in detail from that implied by the original description. This has resulted from several major factors identified during Phase 1 and described briefly below.

The most major change is the deletion of the 4 K word task memory as a major functional unit in favor of a much smaller local memory within each processor module. The tradeoff leading to this recommendation is discussed in substantially more detail later, but the major factors are (1) the questionable efficiency of a task memory in an aerospace software environment; (2) its unquestioned addition of significant numbers of devices and pins to the system which are directly contrary to the reliability objectives; and (3) the problem of providing adequate main memory to task memory bandwidth without resorting to unwieldy (for an ultrareliable spaceborne machine) design techniques such as wide data paths and memory interleaving.

The results of initial subpartitioning studies of the ARMMS modules together with the gathering of the reliability data base are producing conclusions somewhat different than might have been anticipated. First, there is convincing evidence that device failure rates of 10^{-9} failures per hour are being achieved currently in spaceborne applications. Although this failure rate is not projected to continue to decrease rapidly in the coming years, this experience still predicts a gate failure rate of 10^{-11} per hour which is significantly better than had been anticipated initially (by perhaps two orders of magnitude). The implication is that the required degree of subpartitioning and interpartition switching can be less for a given system reliability.

This somewhat positive conclusion is counterbalanced by the somewhat negative results of the subpartitioning analyses. First, the SUMC processor design was reviewed in some detail to attempt to identify reasonable approaches to splitting it into 2 to 4 subpartitions. No approach was found which appears to be acceptable largely because of the high number of interpartition pins required. Next, with regard to memory subpartitioning, it appears that relatively small capacity independent storage modules are preferable to larger, internally partitioned modules both from the point of view of long-term simplex reliability and from the point of view of design feasibility, particularly if other than semiconductor memory is used (e.g., it is difficult to partition and switch internal to a plated wire memory). Finally, with respect to I/O subpartitioning, the I/O unit appears to be very amenable to partitioning by independent channel with each channel having an acceptable failure characteristic.

The net result of these two results is that at present the "best" partitioning approach for ARMMS appears to be a functional one with the optimum switchable partitions being at the level of major system units. Unless this conclusion is overturned in the near future, it would appear that the detailed subpartitioning effort planned for Phase II should be curtailed in favor of more definitive system design and earlier detail design.

A third major factor of significance which has emerged from Phase I is that the previous common assumption of a low ratio of dormant to active failure rate is highly suspect. The value recommended for use as a part of the reliability data base is 0.8 (incontrast with values as low as

0.01 which have been used). This is a highly significant conclusion as it affects the design of long-life computers because it calls into question the general assumption that reliability can be significantly increased by turning power off in spare modules. (It does not remove the significance of power switching for power conservation or electrical isolation purposes.) The proper impact of this result on the ARMMS design has not yet been fully studied.

IV. Mode Flexibility

ARMMS has been envisioned to provide two basic modes of operation: the redundant mode (Mode 1) in which throughput capacity is sacrificed to yield higher reliability in computation and the parallel processing mode (Mode 2) in which the converse tradeoff is made. A third mode (the Simplex mode or Mode 3) is the state to which the computer may degrade as a long-term mission progresses. Although Mode 1 has been presumed to imply some form of TMR operation and Mode 2 presumed to imply multiprocessing, these notions have not been made precise. For example, one could imagine 3 processors executing a critical program in lockstep while several I/O channels were receiving unrelated data from distinct sources. That is, the system need not necessarily be homogeneous with respect to redundancy.

Moreover, for a given maximum number of modules of a given type, there are a large number of submodes which could be identified. For example, if six processors are available, they could be distributed in the following ways:

1. Two TMR machines
2. Three duplex machines
3. Six simplex machines
4. One TMR, one duplex, and one simplex machine
5. One TMR, and one to three simplex machines
6. Two duplex and one to two simplex machines
7. One duplex and one to four simplex machines

In addition, the various redundancy configurations of memory need not necessarily parallel the processor configuration. For example, program data from one memory could be simultaneously provided to three processors whose critical output data could be voted upon prior to transmission to an I/O channel.

Moreover, considering the possible individual restrictions on processors which could be applied (for example, if two parallel TMR configurations were allowed, should all subsets of three processors be able to form TMR machines?), the possible number of identifiable configurations is astronomical. It is clear, both from the hardware viewpoint of interconnections and the software viewpoint of configuration control, that some meaningful limitations must be accepted.

Such limitations should emerge from two factors: 1) those degrees of flexibility which cannot reasonably be envisioned as requirements; or, 2) those which impose the most cost on the hardware and software design.

Two of the eight major sub-modes listed above appear to fall in these categories. First, the requirement for two parallel TMR computers does not seem to be a reasonable design requirement. If a single ARMMS' processor is capable of performing on the order of 1 MIP, then a single TMR configuration of this capacity would seem to be adequate for critical computations. However, the design should allow for all subsets of three processors to be included in the single TMR machine. Second, the ability to support six simplex processors is unwieldy in terms of software control complexity, questionable in terms of need, and uncertain in terms of ability to be programmed reliably. As a result of the Mission Analysis Profile, the recommendation was made that three processors is a manageable upper limit, and this will be used as the baseline value.

These recommendations do not place any necessary limitation on the upper limit of number of processors connected. The number of processors (greater or less than 6) should still ultimately be decided on the basis of total hardware and software costs together with reliability requirements.

However, the maximum of 1 TMR and 3 parallel simplex do suggest limitations on the number of simultaneous data paths which must be provided in the system.

As an additional proposed baseline design parameter, it will be assumed that there may be a maximum of three parallel computers of any redundancy level at any one time. Thus, for example, there may be 1 TMR and two simplex machines, or 3 simplex machines in operation at any time.

The question of whether ARMMS should provide duplex computer capability has not been discussed to date. The fundamental difference between TMR and duplex operation is that TMR provides for failure correction (fault masking) with a high degree of confidence while duplex operation can provide only fault detection. Clearly, TMR is preferred for critical real-time computations for which there is no opportunity to allow time for recovery processes. However, duplexing does provide a complete fault detection approach with less hardware required to be dedicated than does TMR. Thus, for example, one could foresee configuring ARMMS as three duplex processors to achieve maximum computing capacity while retaining fault detection capability for those tasks which are "important" but not "critical". (The distinction between these two criteria is a highly subjective one which could never be adequately measured without reference to a specific mission.)

It would be premature to make a specific recommendation concerning duplex operation in advance of the next phase of the program. However, it is recommended that duplex processor operation be considered for inclusion on the baseline with the maximum number of duplex configurations to be determined as the detailed switch design proceeds.

The baseline capabilities with respect to allowed sub-modes (particularly of the processors) are summarized in Table 1.

The discussion here has centered on processor modes. Suggested baseline memory and I/O modes are discussed in paragraphs vii and viii.

Table 1

MODE FLEXIBILITY BASELINE DESIGN PARAMETERS

Maximum Number of Processors	Possible upper limit of 8 will depend on switch and BOSS complexity, and reliability analysis.
TMR Capability	No more than 1 TMR processor set. Any 3 processors can be utilized.
Multiprocessor Capability	No more than 3 active processors. Any 3 may form the active set. One of the three may be the TMR set.
Duplex Capability	To be considered as the design proceeds. Potential upper limit of three parallel duplex processors.

V. Retention of the Dedicated Executive Module (BOSS)

The suggestion has been made that perhaps we should reconsider the inclusion of a dedicated executive module (i.e. BOSS) in the ARMMS baseline in favor of a floating executive approach. There are a number of reasons why such an alteration might be potentially attractive:

1. A floating executive concept allows all processors to perform executive functions and thereby there is no one module type whose failure cripples the system.

2. A dedicated executive module imposes the added burdens of another module type to be developed and an "extra" module per system.

3. If the executive overhead (e.g. task scheduling, memory allocation, etc.) approaches or exceeds the capacity of the dedicated executive, then the total processing efficiency can be lower than that of a floating concept.

4. With the Floating executive, different processors may in fact simultaneously execute different executive functions.

Hughes has significant experience with the floating executive, and has in fact implemented the concept in two multiprocessor systems - the H4400 and H-3118M. However, in the case of ARMMS, the preponderance of the evidence suggests that the dedicated executive approach should be retained.

1. The development cost of BOSS and its addition to the system is counterbalanced by a decrease in complexity and reliability required for all other processors. For example, configuration control and status monitoring hardware is deleted from every processor. Moreover, if these functions were implemented per processor, it might well be necessary to implement them in an internally redundant manner per processor. Therefore, which approach is the more costly one is not a clear cut issue.

2. The fundamental ARMMS objective of mode flexibility implies a system capability to schedule and implement configuration changes. The problems associated with a processor reassigning its mode role concurrent with monitoring status of all other modules would be difficult to resolve.

3. There are several functions which are simply not amenable to distribution among the processors. These include synchronization, power control, disaster restart, and interrupt reception. It is interesting to note that although the H4400 is

nominally a floating executive system, several functions including the above were centralized.

4. If the Baseline 1 recommendation that no more than three parallel instruction streams be executed at once, then the total executive overhead should be sufficiently low that efficiency is not impaired by queues at the BOSS interface.

5. Attempting to distribute potentially critical system functions may have the effect of enlarging the hard core rather than eliminating it.

Therefore, the original Baseline 0 concept of a centralized executive control module is ratained in Baseline 1. In addition, it is believed that the preferred design approach will be to strictly minimize the functions performed by BOSS to those which are required for fault tolerance and mode flexibility. This approach seems perferable from several viewpoints:

1. Minimizing BOSS complexity will minimize its inherent failure rate.

2. Minimum complexity will lead to the lowest added development and implementation costs.

3. The least added burden for BOSS internal redundancy will result.

4. It will make more feasible the goal of allowing smaller applications to function without requiring a BOSS module.

VI. Recommendation for the Deletion of Task Memory from Baseline 1

In an era when it is fashionable to design increasingly sophisticated memory hierarchies, one runs the risk of appearing foolish at suggesting that a simpler structure is better for a given computer development. However it appears that the weight of the evidence for ARMMS is contrary to the inclusion of a task or cache memory in favor of a small (e.g. 128 to 256 words) local storage unit integral to each processing module. The following paragraphs discuss the considerations leading to this conclusion.

1. Basic Characteristics of High-Speed Storage Alternatives

Some distinctions among possible approaches to providing high-speed storage should first be not ed. One possible set of categorizations is task memory, cache memory, and local storage. The basic distinctions are as follows:

a. A task memory provides for storage of all instructions (and some data) for a given task. Typical size may be 16 K bytes. When execution of a task is to take place, it must first be transferred from main memory to the task memory prior to program execution. It may also require that the previous contents be restored in main memory unless no alteration of task memory contents is allowed.

b. A cache memory implements the assumption that once a reference is made to a specific memory location, the probability that it and nearby locations will be referenced again in the near future is high. Thus, when a task references a memory address, an entire block of memory (e.g. 8 words) is transferred to the high-speed cache. All future program references to this block are then accomplished at the speed of the cache rather than main memory. Various algorithms for deciding which block to discard when a new block is addressed may be implemented (for example see C. J., Conti, "Concepts for Buffer Storage", Computer Group News, March, 1969).

c. The possible usages of local storage in a processor are diverse, and include such functions as instruction retention, instruction look-ahead, index and base registers, local data, scratchpad memory, and general purpose registers.

All of these share the common purpose of reducing the speed mismatch between a high speed processor and lower speed main memory. Or as an alternative statement, they give a processor access to a large capacity, but relatively slow main memory while providing a total speed close to that which could be achieved if all of memory were high speed. Typical speed ratios used are on the order of 10 to 1 for the two memory types. For example, the IBM-360/85 is said to achieve a typical speed equivalent to 81% of that which would be attainable with a single-level storage operating at cache speed, even though the ratio of main memory cycle time to cache cycle time is 12.5 to 1.

Of course, one key motivation for such memories is to maximize the performance/cost ratio. That is, the cost per bit of 1-2 usec core has been lower than the cost per bit of high speed semiconductor memory by a factor of the same order as the speed ratio. Another significant factor is that restricting the capacity, and therefore the volume, of high-speed storage reduces propagation delay problems associated with the large physical dimensions of commercial machines.

2. Some Performance Comparisons of Task and Cache Memories

Task and cache memories have substantially different characteristics which can be categorized by the relative advantages given below: The advantages of a task memory include these: 1) Some aspects of the hardware implementation are simpler. For example, no "hit-miss" logic is required since all instructions associated with the task are known to be in the task memory, and no decision process concerning which block to discard is required. 2) Execution time within a task memory is less subject to uncertainty than with a cache memory and performance is independent of addressing patterns. 3) In a multiprocessor system, the task memory may present fewer difficulties in the event of a failure or interrupt. That is, since task memory contents are relatively static during task execution, the state of the system is easier to know and accommodate.

The relative advantages of the cache approach include these: 1) The use of the cache is transparent to the user. Memory appears to him as a single-unit. 2) The cache does not limit task size as does the capacity of task memory. 3) The efficiency in use of main memory is inherently better with a cache (if they are of equal capacity). The task memory must load all instructions associated with a task while the cache loads only those which are most likely to be executed. 4) The efficiency of a cache is relatively insensitive to task size.

3. The Inapplicability of the Task and Cache Approaches for ARMMS

In evaluating the above characteristics and relative mertis, we believe that a very strong statement can be made: All of the underlying assumptions and objectives of the task or cache memory approaches are questionable with respect to spaceborne multiprocessors in general and ARMMS in particular.

a) The cost effectiveness of task and cache memories depend on high cost ratio between them and main memory. This phonomenon does not occur with respect to plated wire and semiconductor memories (probably the two primary ARMMS candidates) particularly of flight-rated quality.

b) Their performance effectiveness depends on a high speed ratio with respect to main memory. Again, this property does not characterize the probable ARMMS choices.

c) The propagation delay problems averted by reducing the proportion of memory which is high speed are less of a factor for a spaceborne machine which must emphasize packaging density as a matter of first priority.

Other requirements of an effective task or cache are undesirable for ARMMS. Perhaps most notable is the sheer addition of devices, pins, watts, cubic inches, and failures per hour, which they imply. This penalty is large in the case of a multiprocessor where the ratio of task memories to processors is at least one-to-one. Another critical problem would be that although the main memory cycle time may be longer than task memory cycle time, the cumulative data rate into the task memory from main memory must approximately equal the data rate between task memory and processor. This is typically accomplished by the use of wide data paths and interleaved data from main memory to the task or cache memory. (It is in this way that the speed mismatch is resolved) For example, the IBM 360/91 employed 8 or 16-way interleaving with a data path width of 64 bits. Both of these attributes are undesirable for ARMMS because of the multiplicity of connections involved and the difficulties in reconfiguring and voting upon interleaved memory.

Based on the above factors, we conclude that neither the task memory nor the cache memory approaches are sufficiently promising to be retained in the ARMMS baseline, and we recommend that the concept of a small, local store to be explored as described in the following section.

4. Applicability of the Local Storage in the ARMMS Processor

A prime source of inefficiency in multiprocessing is contention for memory access by theprocessors. Seve ral approaches involving use of relatively small amounts of storage in the processors to reduce the number of main memory accesses may be employed. Of course, the use of general registers is by now a classical technique which allows intermediate operands to be retained internal to the processor.

Two other basic local store uses can reduce the number of memory accesses required. One is to retain the previous n instructions, and should a branch backwards to within this bound occurs, then no main memory instruction access is required. This technique may typically reduce the number of instruction accesses by 4% if $n = 8$. More significantly, instruction retention speeds access to perhaps 35% of branch instructions. (These quantities are based on detailed analysis of an extensive set of aerospace programs.)

Note that instruction lookahead, although a useful processor speed-up device, actually increases memory accesses required because some instructions accessed will not be executed because of branches.

The second additional opportunity to reduce memory bandwidth through local store is to provide the ability to access and retain data most frequently used by a given task. This requires the software discipline of identifying by task which data should be pre-accessed and stored. Although this would not be a popular requirement with programmers, the potential payoff is great. For example, if 64 local data locations are provided, they may be used for up to 60% of the data references to memory. Note that this technique is the converse of the task memory approach in which instructions are pre-loaded and data is accessed as needed.

Local storage for these functions has the following advantages in ARMMS:

1. Total memory accesses required per processor for a given task are reduced by perhaps 15%.
2. The amount of hardware added per processor is small relative to task or cache memories.
3. The amount of processor-oriented data which must be considered upon interrupt or failure is reduced.
4. The local data area has potential as a temporary location for accumulation of output data which may be a usable adjunct to the memory data protect system.

VII. OVERVIEW OF THE BASELINE 1 ARCHITECTURE

1. Module Classes - Baseline 1 contains four basic module classes, processors, main memory modules, I/O units, and BOSS. Relatively little intramodule switchable subpartitioning is foreseen except for the internal redundancy which will be required by BOSS. Also, the Baseline 1 memory includes spare bit planes which can be utilized after an internal array failure. To date, internal fault detection within the other module classes has not been considered in detail. This activity is scheduled primarily for Phase II of the program. Bulk memory is considered to interact with the system via the I/O units.

2. Maximum Number of Modules Per Module Class

The following upper limits of number of modules per module class will be used as Baseline 1 values.

- Up to 8 processor modules which at any one time may be configured into no more than 3 parallel processors (one of which may be in TMR configuration).
- Up to 32 memory modules of varying capacities and technologies. All will utilize common interface designs for connection to the other module classes. An overall approach to memory data protection will be specified later, but initially it is expected that the ability to write into modules which

contain instructions will be under strict control of BOSS.

- o Up to 4 independent I/O units capable of accommodating a cumulative 10^7 bits per second data rate.

- o There is one BOSS module which achieves high reliability through internal redundancy. One important design goal is to design the system so that smaller applications do not require a BOSS module. Major implications of the goal are that such systems would sacrifice the reconfiguration capability afforded by BOSS and that the processor module design must accommodate basic executive functions.

3. <u>Module Interconnections and Data Paths</u>

The following major data paths are defined.

- o Memory Data to Processor and BOSS - Several busses of sufficient width and speed to support three parallel processors are provided. As subsequently discussed, the speed of each line must approximate 15×10^6 bits per second.
- o BOSS, Processor, and I/O Data to Memory - The speed of this path must be comparable to that of the memory output data signal paths.

- Processor to I/O Control and Data - The processors use this path to initiate I/O operations and as an alternate path for critical control signals to the outside world.

- Memory to I/O Data - This path is specifically identified as a tentative one. Its independence of the memory to processor data busses is prompted by the lower data rate required to sustain I/O, and the greater requirement to vote on memory to I/O data than on memory to processor data (this latter statement is itself an arguable one). The primary reason for including this path in Baseline 1 is to stimulate the trade-off between voter complexity, circuit speed, total system pins, and number of source and destination modules.

- BOSS Control and Response Bus - This is the path by which BOSS interrogates and controls the remaining modules and samples their internal status. A sequential, polling approach is suggested as described later.

4. <u>Voter Placement and Redundant Mode Operation</u>

Placement of voters in ARMMS may have a critical impact on reliability and cost. Insufficient use of masking redundance may fail to provide the desired critical mission phase reliability while overzealous application could produce a cumbersome system with too much sacrifice in cost and speed. The approach suggested here is a first attempt at identifying the necessary balance.

The guidi ng principles adopted are the following:

1) All data which is sent to the outside world must be voted upon at two distinct physical locations and at distinct times. This is desirable to help to protect the system from transient faults and permanent voter faults. 2) All output from the processor either to memory or to I/O can be voted. This provides the most complete fault detection and correction for processed data.

Under these principles, voters are postulated as follows: 1) From processors to main memory; 2) from processors to I/O; 3) from main memory to I/O; 4) Between the external word and I/O units (for both input and output data).

Voting capability is not postulated as follows: 1) From main memory to processors and BOSS (except perhaps internal to BOSS); 2) from BOSS to other modules (except perhaps internal to BOSS); 3) from I/O to main memory; 4) from modules to BOSS.

A brief discussion of how the system is visualized to operate in TMR mode is in order. BOSS would identify that a critical phase or task is to be accomplished and would schedule any three operational processors to begin TMR operation. All instructions and data would be provided synchronously to the three processors. This will require either that a single memory module can output data to three processors at once or that the programs be triplicated in storage. A possible goal would be to provide the capability for either technique with the choice to be made based on mission requirements. The second approach may have more difficult hardware and software implications because the three processors would be addressing different memory addresses and would therefore have differing states of address generation logic.

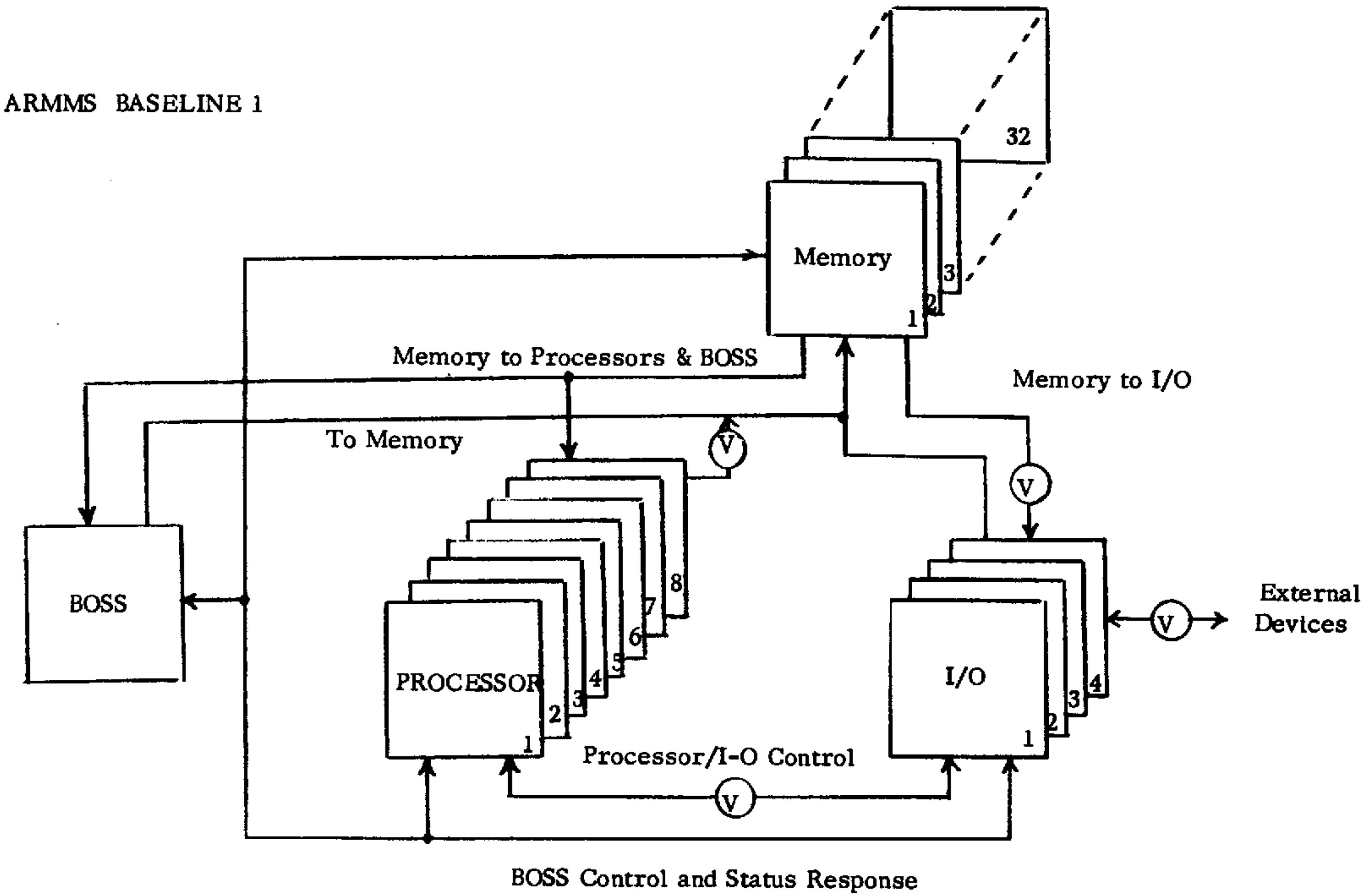
Then as the processors execute instructions and store data in main memory, all such processed data would be voted upon. Again there is the choice of whether to return the data to one or to three memory modules with corresponding hardware and software tradeoffs to be studied. Another possible approach would be to implement TMR programs to store all I/O data sequentially into three modules. The number of active voters will differ for the various approaches.

When an I/O operation is to be initiated, this action would be voted at the Processor/IO control interface. Here it is mandatory to be able to activate three I/O units. Both I/O commands and short data blocks may be transmitted via this interface. Three active voters are probably required at this interface.

The processors may activate three I/O units to transfer large blocks of data via memory. The notion of voting on memory output data to I/O depends on whether the data has been triplicated in memory from the processors. If not, the value of memory to I/O voting is brought intc question, and critical output results should be transmitted directly from the processor and voted again at the output of the I/O unit.

The Baseline 1 architecture as described in this section is shown in Figure 1. Possible characteristics of the system data paths are given in the following section.

ARMMS BASELINE 1



VIII. POSSIBLE CHARACTERISTICS OF THE MODULE INTERCONNECTIONS AND DATA PATHS

1. Processor/Memory Interface Composition and Speed

The maximum intermodule data flow rate occurs between processors and main memory. The combination of target processor speed, word length, and number of parallel processors have the primary effects on the speed required of the interface with added features such as error correcting code bits and memory protect bits having a lesser effect.

As a sample case, consider the following parameters.

1.6	memory accesses per instruction
1.8	million instructions per second per processor
3	processors maximum active

Processor Output/Memory Input Data

32	bit data word
20	address bits (1000 K words directly addressable)
16	control, memory protect, error codes, etc.
68	Total

Memory Output/Processor Input

32	bit data word
12	status, error codes, etc.
44	Total

Then for these figures, the data rate which must be accommodated by the interface design is given by

$R_T = 1.8 \times 10^6 \times 1.6 \times 3 \times (68+44) = 9.45 \times 10^8$ bits per second.

A primary design requirement will be to minimize the number of interface pins consistent with feasible intermodule data transmission rates. Thus, if we can achieve an intermodule signal rate of $15x10^6$ bits per second, then this suggests the total number of signal lines at the processor/memory interface can be on the order of $9.45x10^8 \div 15x10^6 = 63$. This is approximately one-sixth the total of the processor/memory input and output signals 3x(68+44) = 336. This suggests that the signal lines can be time multiplexed. For example, the processor output lines could perhaps be limited to 12 in which a single memory data request would require approximately six 12-bit transmissions while a response from memory might utilize 6 8-bit byte transmissions.

All of these computations are very coarse, of course. For example, as a second order refinement, requests for instructions need not be accompanied by a processor output data word. This fact alone could reduce the total data flow by about 15%. Conversely, if a simultaneous TMR, two simplex processor interface is required with full two-way voting in the TMR case, the number of simultaneous paths increases to 5 from the assumed 3. Also, the baseline requirement of 10^7 bits per second of I/O data has not been included (although this represents less than 10% additional data flow in the system). At least one other design decision will have a major impact on the total memory/processor data flow, and that is the speed of the processor. The capability of 1.8 MIPS (itself a coarse number) represents a probable maximum, but this number could decrease toward the present SUMC processor capability.

However, the case presented is stringent enough to allow the following conclusions to be drawn:

1. A bit transmission rate of 15 MHz is a reasonable upper limit of the memory/processor signal interfaces.
2. An interface design can be achieved which requires less than 30 pins per processor/memory path, with 20 as a potentially reasonable design goal. The total number of pins per module to support this interface would be a multiple of this number. That multiple will be the maximum number of memory/processor busses which will be a minimum of two and a maximum of perhaps five.

2. BOSS/Module Interface Composition and Speed

The overall functional requirements of BOSS will be a subject of continuing study over the next nine months. However, as a minimum, BOSS will control system configuration and status monitoring. Thus it must be able to sample status of all system modules and to connect or disconnect any module either by power switching or other means. Whatever approach to system synchronization is adopted will also almost certainly be a BOSS function.

As discussed elsewhere, the preliminary baseline design approach to BOSS is to minimize the functions assigned to it and to minimize its internal complexity. One key means to this end is to allow BOSS to have direct access to main memory via the processor/memory busses. That is, BOSS will be connected to main memory in exactly the same fashion as is a processor, thus eliminating the requirement for a large amount of internal BOSS storage no matter what functions are ultimately assigned to BOSS.

As with the other module classes, pin minimization appears to be a key requirement for BOSS. Thus, a time-multiplexed control and response bus is incorporated in Baseline 1. The normal operation of this bus would be in polled fashion with each module (including voting networks) interrogated sequentially with each being required to reply concerning its internal status. The number of messages per polling cycle would ordinarily be the sum of the number of processors, I/O units, memory modules, and voting networks. A safe upper limit for this number appears to be 64. The bus width and speed is determined by this number together with an acceptable system sampling interval. For example, if 2 milliseconds is an acceptable interval, then the module sampling interval could be about 30 microseconds.

In a normal (i.e. non-failed) environment, BOSS would output a module address followed by an interrogate status command word. The addressed module would then respond with a verification of its address followed by its internal status data, which normally could be as simple as an "all-is-well" signal. Failure to respond would of course be a failure indication to BOSS. Thus, the normal message flow in the 30 usec interval could be as little as about 30 bits of information. Thus, this bus can be very narrow in data width or very slow (i.e. 10^6 bits per second), or both.

This bus can also provide several other important functions. For example it provides the means for BOSS to control mode changes, by allowing it to command a processor to suspend activity at the end of its current task for reassignment to a TMR role. Further, it is the mechanism for controlling reconfiguration of spare memory bit planes if they are included and control of other module-oriented fault control features. Also, it provides a direct BOSS to I/O data path to allow, for example, BOSS to communicate its own internal status to the ground via telemetry or for control of disaster restart from the ground. Also, the BOSS control bus would be used to transmit system-wide commands to all modules or to all modules of a given class.

This concept impacts the design of the other modules. One notable case is that if a module detects an internal fault it should ordinarily suspend activity until BOSS interrogates it and issues further commands (such as power off or retry or roll-back to a specified location).

Two other observations concerning the control and response bus are that 1) the data rates are low enough that the use of bidirectional lines can be considered and 2) the bus itself requires replication for reliability reasons.

3. Memory Data to I/O

As noted earlier, the purpose of including a dedicated memory to I/O data interface at this time is to allow more detailed exploration of voter placement and its implications. An important objective would be to minimize the number of voters required by time multiplexing the data flow to the I/O units. Thus if a peak I/O transfer rate in the redundant mode is 2.5×10^6 bits per second, and if a single line can accommodate 15×10^6 bits per second, then all I/O data could be transmitted to the I/O units in bit serial fashion, using only three voting networks.

Overall, the interface might require the following functions:

Memory output data to I/O:

32 bit data word,

12 status, error codes, etc.

I/O to Memory

20 address

16 control, memory protect, error codes, etc.

Assuming signals in both directions are heavily time-multiplexed, a total of 10 pins per interface may be a reasonable goal.

4. Processor to I/O Control and Data

The comments concerning minimization of voters by time multiplexing of data mode above are equally applicable to the path from processor to I/O. The speed required for this interface can be very low because of the infrequency of execution of I/O instructions and direct processor to I/O data. The information crossing this interface will include the following:

Processr to I/O Unit
- 32 bit data words
- 32 bit command words
- 12 status, error code, etc.

I/O to Processor
- 4 status, acknowledge, etc.

A total of 6 pins per interface is a reasonable goal.

5. Preliminary Module Pin Count Estimate

Based on the preceding discussion, we can tabulate an estimate of the total interconnecting pins which will be required at the module interfaces. These estimates depend on the number of pins per interface type and the number of interface types per module implemented for reliability purposes. Although the entries in Table 2 are little

MODULE TYPE

INTERFACE	PROCESSOR			BOSS			I/O UNIT			MEMORY		
	Pins/Bus	Busses	Total	Pins/Bus	Busses	Total	Pins/Bus	Busses	Total	Pins/Bus	Busses	Total
Memory Data to Processor & BOSS	10	3	30	10	3	30				10	3	30
BOSS, Processor, & I/O Data to Memory	20	5	100	20	5	100	20	3	100	20	5	100
Processor to I/O Control & Data	6	4	24				6	4	24			
Memory Data to I/O							10	4	40	10	4	40
BOSS Control & Response	10	2	20	10	2	20	10	2	20	10	2	20
I/O to External							10	4	40			
TOTAL PINS PER MODULE			174			150			224			190

TABLE II. Module Pin Count Estimate

better than assumptions at this time, they do serve as a starting point from which the design may evolve. As shown in the table, the estimates of number of pins per module range from 150 to 224. These estimates do not include certain functions including power, power switching, clock inputs, and ground. Although these pin estimates are not small, they are within the bounds allowed by the sample module package shown on page 2-11 of Hughes FP 71-11-212. However, an increase by a factor of 2 over these estimates will potentially limit the package size which can be achieved.

Another preliminary estimate of interest which may be made is the number of voting networks required (exclusive of those internal to BOSS). As an initial basis, assume there is one voting network per pin in those paths in which voting can occur, as shown below.

Data Path	Number of Voters
BOSS Processor and I/O Data to Memory	100
Processor to I/O Control and Data	24
Memory Data to I/O	40
I/O to External	40
Total	204

Whether or not this is a manageable number depends on the design of the voter as it emerges over the coming months.

SECTION 3

MISSION ANALYSIS PROFILE

TABLE OF CONTENTS

Preceding page blank

TABLE OF CONTENTS
(continued)

LIST OF TABLES

LIST OF TABLES
(continued)

LIST OF ABBREVIATIONS

ARRMS	Automatically Reconfigurable Modular Multiprocessor System
OFF	Orbital Free Flight
OPF	Orbital Powered Flight
MAP	Mission Analysis Profile
NA	Not Available
N/A	Not Applicable
NE	No Equivalent

SECTION 3

MISSION ANALYSIS PROFILE

Since ARMMS is intended for multiple mission-types, it is essential that its design encompass the range of mission performance characteristics which can be identified. To identify such characteristics, more than thirty sources of historical data and future projections relating to space computer characteristics were studied. The result is the Mission Analysis Profile report which follows.

One of the major areas of interest was to estimate the peak computing capacity which should be sought. The conclusion was made that for a computer system intended for use in 1978 or later, a capability in excess of 4 million instructions per second should be a realistic objective. It should be observed that most missions reviewed project a much lower requirement. Tables 3-1 and 3-2 summarize speed, memory, and I/O requirements ranges as they were extracted from available sources.

Of perhaps more immediate value to the ARMMS design were the MAP studies of detailed task profile characteristics which should be anticipated. Sixteen major tasks were identified, and task timing and criticality parameters were established for each (see Table 3-5). These parameters included restart timing sensitivity, interruptability, failure criticality, memory redundancy, and task duration.

A detailed analysis was made of task characteristics of the boost phase application modules of the Saturn V flight program. Forty-two tasks were identified and detailed to the level of number of instructions, iteration rate, interrupt conditions, task precedence relationships, etc. (see Table 4-1).

1. INTRODUCTION

1.1 Purpose

This report documents the data gathered during Task 1 of Contract No. 7-104518-R-D7 (Hughes Aircraft Company, Fullerton, California). This data is to be used to aid in the design of the Automatically Reconfigurable Modular Multiprocessor System (ARRMS) under contract to NASA-MSFC. The data was extrapolated from historical data and future projections of Spaceborne software.

1.2 Document Organization

This report has been organized in three principle sections, briefly discussed below.

1.2.1 Spaceborne Software Characteristics

Section 2 of this report briefly describes dominant and pertinent characteristics of Spaceborne software. Summarized numerical data, extracted from Section 3, General Mission Analysis Profile, is presented to support the description where applicable.

1.2.2 General Mission Analysis Profile

The heart of this report is Section 3, the General Mission Analysis Profile. It is this section where the MAP (Mission Analysis Profile) parameters defined during this task are documented.

For the sake of clarity, these parameters have been organized into a number of separate subsections, designed to distinguish between hardware and software dependent parameters, and to distinguish between the computational requirements of discrete tasks versus those of complete mission phases.

1.2.3 Saturn V - Boost Profile

The data reviewed for the General Mission Analysis Profile is on such a level that many interesting details underlying the MAP parameters are left to the imagination of the reader. To highlight such detail, a detailed profile of Saturn V (Boost) operational software is provided in Section 4.

This profile may be interpreted as characteristic of most of the tasks described in Section 3, General Mission Analysis Profile.

2. SPACEBORNE SOFTWARE CHARACTERISTICS

This section briefly describes those characteristics that may be pertinent to the ARRMS hardware/software architecture.

It should be realized that most of these characteristics are extracted from the design normally applied to spaceborne applications. That is, different (unknown) designs may result in different characteristics. It should also be pointed out that, although some of these characteristics are factual (based on current experience), some of the characteristics are based on currently unimplemented designs and are, therefore, subjective extrapolations of future requirements.

2.1 Process Structure

The spaceborne software encompasses, at least, processes with the activation and execution characteristics described below.

2.1.1 Process Types

Synchronous Processes

These are processes that are executed repetitively at a specified rate, or at specific time intervals, over a period of time. A wide variety of repetition rates have been noted. The most common ranges fall between .1 - 100 per second. However, repetition rates as high as 400 per second have been proposed for specific mission functions.

Accuracy requirements for the time interval between iterations are not normally available; and where available, they are somewhat arbitrarily chosen. Generally, the required accuracy is proportional to the frequency; high for high repetition rates, low for low repetition rates.

It is, of course, standard practice to maintain as high an accuracy as possible, and verify that transient disturbances have little or no effect upon the system performance.

Asynchronous Processes

All other processes can be categorized herein. These are, therefore, processes that are not executed at specific time intervals over a period of time.

Some of these processes are considered time-critical. This means that they have to be executed within a "very short" time interval of the point at which the need for their execution has been detected. This time interval may be as short as a few milliseconds. The need for time-critical processes is generally imposed by the requirements of the application. However, it is also sometimes imposed by the (hardware) interface between the application software and the application, rather than the application. As the existence of time-critical processes significantly affects the performance and complexity of the software, it should be a factor to be considered for minimization in any system interface design.

2.1.2 Activation Conditions

The conditions upon which activation of synchronous and asynchronous processes depend are varied and sometimes complex. Any logical combination of the following events may form the activation conditions--the reasons to start or stop a process:

- o Start/Stop of other process(es)
- o Value of data item(s) or combinations of data items
- o Discrete events - signalling conditions within the environment
- o Values of sensor data
- o Elapsed time within mission or phase, etc.
- o Elapsed time interval after occurrence of other event

A decision pertinent to the ARRMS design is whether detection of and reaction to any or all of these events should be an integral part of ARRMS or should be performed within the application software.

2.2 Data Base

For the purpose of this discussion, all information items that do not represent instructions, and are stored in either main storage or in bulk storage, are considered part of the data base. Data base handling techniques associated with error detection/recovery are not discussed in this paragraph but are touched upon in Paragraph 2.3 - Error Detection/Recovery.

2.2.1 Data Base Structures

In current implementations and current designs of aerospace software, this is a trivial subject. It would be a non-trivial subject, however, for large Space Stations where many extensive data-inter-related experiments are performed and other large information bases are required. Nevertheless it remains a highly application (mission) related subject. As such it should have no effect on the ARRMS design, and will not further be considered in this study. Note that there is no reason to believe that the pertinent structures of large data bases will look any different from ground-based systems.

2.2.2 Data Base Sizes

The analyzed mission profiles indicate the following requirements for Data Bases:

	Main Storage (words)		Bulk Storage (words)	
	Min	Max	Min	Max
Tasks not including experiments	4K	44K	0K	16K
Experiments	12K	45K	22K	10M

2.2.3 Other Characteristics

The Data Base for spaceborne software contains more than just those data items that are global to the processes. Data items global within the processes are generally also found in the Data Base. In fact, there is a strong tendency to retain all data items within the Data Base, regardless of their scope. The Data Base then is that part of storage that contains all data items that are used during a mission and are accessible to all processes.

The reasons for this approach are many. The most significant reasons are discussed below.

(1) Initialization - This should be a centrally performed (and controlled) function to, first of all, have a positive control that all data is indeed initialized and is initialized to the proper value.

Secondly, many of the data values are mission and/or phase dependent and, therefore, unknown to the software designer at the time the software is designed. Lastly, it aids in the ease of restarting the processes from the start of mission or start of phase.

(2) Parameter communication - Direct transmission of parameters between processes is not always practical because of the following reasons:

- o Activation conditions of other processes are frequently not known to a process
- o The processes requiring or generating the parameters may change with mission or time within mission
- o Multiple processes may affect or use the parameters

It is, therefore, often necessary for a process to access/write its parameters without any knowledge of which processes it actually communicates with.

(3) Data Protection - Retaining all data items within a centrally available block of storage significantly simplifies protection of the integrity of the data. In fact, reasonable data protection methods are practical only on that basis.

- o Erroneous access to non-data

  Errors within the processes that may result in erroneous modification to stored instructions can be prevented because it is known which areas of storage do not contain data and therefore should not be accessed.

- o Erroneous access to data

  By centrally assigning and controlling access rights to the data items, unauthorized access to data items by the software modules can be, reliably, prevented.

- o Prevention of read/write conflicts

  Any system, in which processes (accessing common data) progress concurrently, contains potential read/write conflicts. Any practical method to fully prevent this requires that the data allocations are centrally known.

2.3 Redundancy and Recovery

2.3.1 Tasks

The Mission Analysis Profile (Section 3) does not directly relate the tasks to redundancy modes. It does not do so for the simple reason that redundant execution is not a requirement inherent to a task. It does, however, indicate for which tasks erroneous outputs must be detected (D) prior to propagation and for which tasks this is unnecessary (N). Note that this only identifies a predominant requirement, not a universal one. A finer breakdown into subtasks and/or subphases is likely to show some variety within a task. It is, of course, generally true that D-classified tasks require redundant execution or redundant hardware, simply because no other satisfactory method has been found.

A second, pertinent, parameter identified in Section 3 is the restart timing sensitivity of the tasks. It again only indicates a predominant requirement. It also assumes that the task must be restarted and indeed can be restarted.

If, for example, a vehicle has a "safing mode" and does not have to perform any critical maneuvers, the resumption of the various tasks could be allowed to take a considerable amount of time. This has, also, a direct bearing on the restart techniques that could be utilized. System checkpoint/restart may be perfectly acceptable for some applications where a long period of time is allowed for recovery. However, for other applications, it may be necessary to restart an individual task or subtask rather than the whole system. The latter would put a restriction on the systems design in that inputs to a task must be preserved and outputs must not affect anything until a task has been successfully completed.

2.4 Data Bases

There is in theory no data that cannot be recreated by reload from the ground, recalculation, reinitialization, or other backups. Whether a Data Base (or a part of it) should be maintained redundantly is primarily dependent upon the restart timing sensitivity of the tasks using it. Where the tasks do not place any critical requirement upon the data base recovery, it becomes significant which means of recovery are are indeed available during the mission. Redundancy and recovery of data bases is thus directly dependent on the systems design and, therefore, highly mission dependent. The dominant requirement is, however, that it must be possible to maintain complete integrity of a large data base. This not only means that the data has to be stored redundantly, but also that failures occurring during task execution do not affect the data base.

2.5 Processor Loading

To establish whether there is a need for the use of multiple processors for the parallel execution of mission tasks, the "Processor Requirements Summary" (Paragraph 3.1) is used. The summary indicates a maximum requirement of 1.732 MOPS. It is not unreasonable to assume that the available MIPS should exceed the computed MOPS by 40 percent of the total capacity. This allows for growth and unanticipated inefficiencies in the currently planned missions. Thus, for planned missions a capacity of $\frac{100}{60} \times 1.732 \approx 2.9$ MIPS is required.

It is not unrealistic to assume that the requirements for missions not currently analyzed could exceed this by fifty percent. Therefore, for a computer system meant to be used over at least the next ten years, 4.4 MIPS should be a realistic objective.

We can roughly calculate the minimum single computer capacity required within a multiprocessor system based on the following assumptions:

(1) It is impractical to assume that more than three processors can be effectively used to perform parallel processing of interrelated processes.

(2) Adding a computer to a configuration only adds 80 percent of the apparent single processor capacity.

This results in a three computer configuration having a maximum capacity of 2.52 times the capacity of a single computer. One computer should, therefore, have a processing power of $\frac{4.4}{2.5} \approx 1.8$ MIPS. This is realizable within a single computer, but about ten times the power necessary for the smaller mission requirements reviewed.

2.6 Adaptability to Multiprocessing

Anytime the use of multiple processors for an application is proposed, there is an immediate and real concern of how effective this really is. It is nearly impossible to answer that question in general terms. It is easier to describe under what conditions the use of multiple processors becomes ineffective.

First of all and most obvious, is the fact that if an application consists of one process module, it cannot be multiprocessed. In other words, an application must be divided in modules.

Secondly, if all modules are linked together in a sequential relationship such that each module has not more than one successor, the application cannot be multiprocessed.

The point that we are leading up to by these obvious statements is that, to obtain reasonable effectiveness of the multiprocessor system, the application has to be designed to run on a multiprocessor system. This design frequently results in a significantly different modularization than that designed for a single processor. Sequential relations should be minimized; cooperative (parallel) relationships should be emphasized. Fortunately the structure of the spaceborne software (multiple synchronous processes executing at different repetition rates) is such that it is naturally amenable to the modularization design indicated above.

The next obvious problem is that n processors will never execute an application n times as fast as a single computer. In other words, it is generally impossible to design the module such that the loading of the n computers is nearly equal. How bad the difference in loading really is, is purely dependent on the particular application, and the cleverness of the application software designer.

The last point to be described, and the only point that may not be obvious, is the effect of "Richard's Anomalies". We will not completely describe these anomalies here, but only describe one practical implication.

It is natural that the execution time of the individual software modules may differ between repetitive executions. The effect of "Richard's Anomalies" is that the total execution time of a set of modules on a multiprocessor system varies disproportionally to the variation of the execution time of individual modules. As a matter of fact, if the execution time of a single module is slightly decreased, the total execution time can significantly increase. The upper bound on the increase on a three processor system is 67 percent; far too large to be practically acceptable.

It is significant to note that both the last two problems are minimized by making the individual modules as small as possible or making the modules as large as possible.

To summarize, spaceborne applications can usually be divided into cooperating, parallel, modules. However, the resulting effectiveness of the multiprocessor system is directly dependent on the cleverness of the design of the application program, as well as the design of the Control Executive.

3. GENERAL MISSION ANALYSIS PROFILE

The parameters which specify or define the range of computational capabilities required by an advanced avionics data management and control system are a function of a number of discrete but interrelated mission characteristics. In particular, for space oriented missions, such as those reviewed during this study and those anticipated for the ARMMS application, the key parameters which define the level of computational power required are directly dependent on:

- The individual jobs or tasks to be performed such as guidance, navigation, redundancy management, etc.
- The mission phases and subphases
- The task timing and criticality requirements

As a result, the parameters presented in this section which constitute the defined MAP have been organized into tables which group separately those parameters which are task oriented from those which are phase dependent. In particular, the MAP consists of five specific tables:

(1) Overall Processor Requirements Summary (Table 3-1)

(2) Phase Hardware Parameters (Table 3-2)

(3) Software Structural Parameters (Table 3-3)

(4) Task Hardware Parameters (Table 3-4)

(5) Task Timing and Criticality Parameters (Table 3-5)

Figure 3-1 defines the interrelationship between the individual MAP tables which are discussed in detail in the remainder of this section.

In order to simplify the MAP from the viewpoint of the casual reader, the MAP tables are presented in the sequence listed above and reflected in Figure 3-1 from top to bottom. However, the progression of definition of the MAP parameters was essentially in the reverse sequence and the serious investigator may find it more meaningful to begin at the task level and work backwards through the phase tables to the overall processor summary.

MAP COMPONENT STRUCTURE

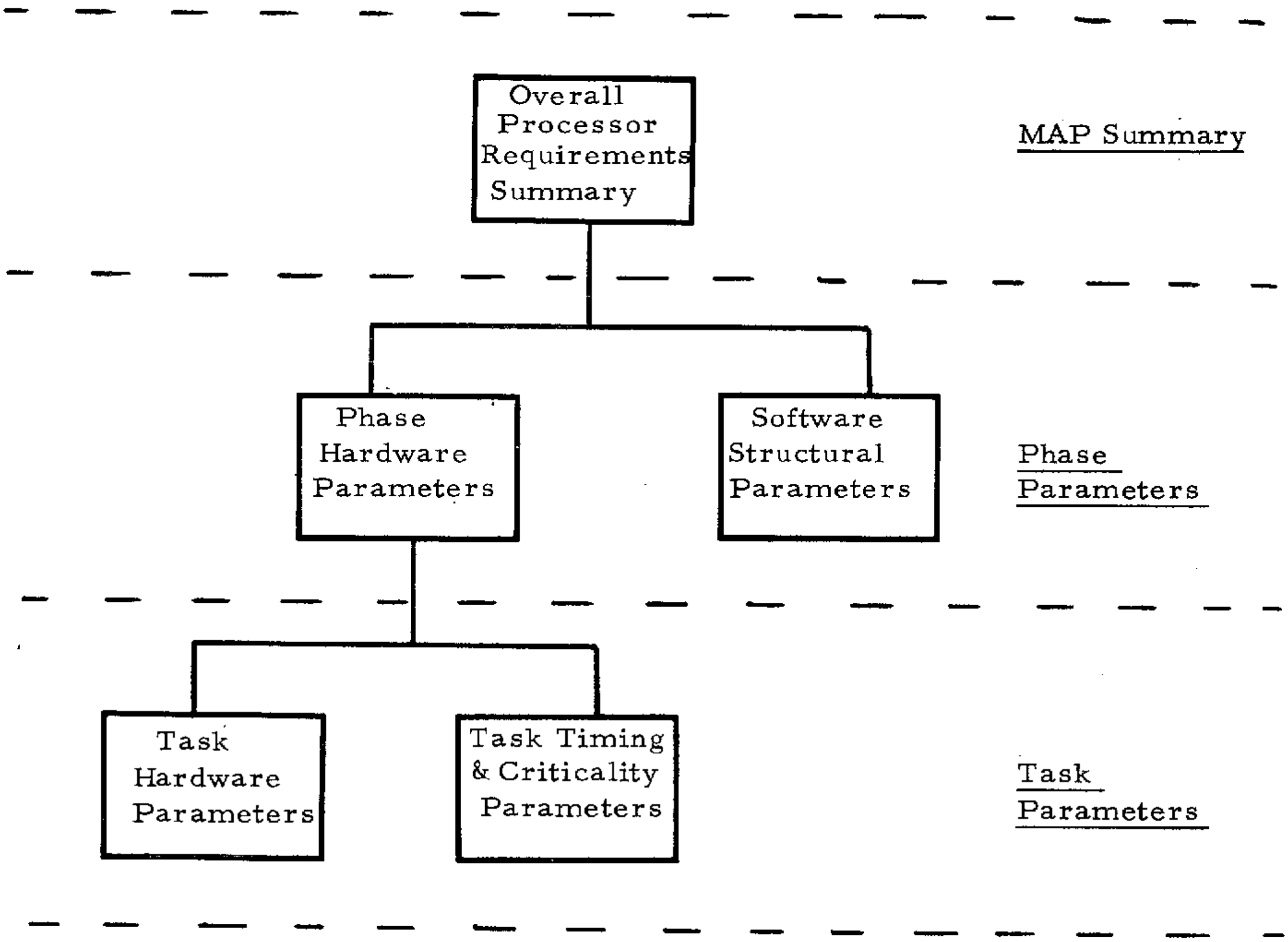


Figure 3-1

In general, each of the parameters defined in the MAP tables has associated with it two values, a minimum value and a maximum value. This defines the range of variation of the quantity being defined as reflected in the reference material.

For each of the parameters defined in the MAP tables, there is an associated footnote defining the source of the value defined. The footnotes are identified by the subscripts in parentheses and are listed in Appendix 1.

3.1 Processor Requirements Summary

Table 3-1 presents the overall processor requirements summary. In general, the values presented are condensed or extracted from the phase summaries of Table 3-2 in the manner specified by the appropriate footnotes.

In order to clarify the presentation, the overall processor requirements have been sub-divided into three (3) categories: Boost/OFF/OPF, Interplanetary, and Experiments; and two totals: one for mainframe core only, and one for both mainframe and bulk memory requirements.

While Boost, Orbital Free Flight (OFF), and Orbital Powered Flight (OPF) are mutually independent phases, there is considerable overlap in function between them. Most of the appropriate references reviewed made a clear distinction between these phases and the interplanetary phase. It is for this reason that they are presented in Table 3-1 as two separate categories. The Experiments data was grouped into a separate category because the parameter data varied over such wide ranges that they completely mask the requirements of the other categories and obscure the true picture if only the totals are presented.

3.2 Phase Profiles

Table 3-2 presents the phase hardware parameters summaries. The phase hardware parameters are defined to be those parameters which are mission phase or subphase dependent and which have a direct influence on the processor hardware characteristics; i.e., Word Length, Memory Size, Execution Speed, etc.

Five (5) primary mission phases have been defined in this MAP:

OVERALL PROCESSOR REQUIREMENTS SUMMARY

MEMORY REQUIREMENTS

Parameters	Instructions		Data (words)				Total Words	
			Mainframe		Bulk			
	Min	Max	Min	Max	Min	Max	Min	Max
1) Boost, OFF, OPF	13460 (43)	74526 (43)	5530 (43)	43748 (43)	None (45)	115979 (43)	18990	234253
2) Interplanetary	7690 (44)	47799 (44)	4490 (44)	22724 (44)	None (25)	8000 (44)	12180 (44)	78523 (44)
3) Experiments	21450 (49)	97280 (52)	12550 (49)	44600 (50)	22000 (48)	9772000 (18)	56000	9893000
Totals (Mainframe core only)	36800 (51)	183564 (51)	19185 (51)	93938 (51)	N/A	N/A	55985	233640
Totals (Worst Case)	36800 (51)	183564 (51)	19185 (51)	93938 (51)	22000 (51)	9889945 (51)	77985	10146559

N/A = Not Applicable

Table 3-1

OVERALL PROCESSOR REQUIREMENTS SUMMARY

PROCESSOR CAPABILITY

Parameters / Categories	(K) Ops/Second		Word Length (Bits)		Instruction Mix (%)		I/O Requirements Bits x 10^3/sec	
	Min	Max	Min	Max	Add	Mult.	Min	Max
1) Boost, OFF, OPF	74.3 (43)	1732.0 (43)	16	32	87 (45)	11 (45)	350 (46)	800 (46)
2) Interplanetary	285.454 (44)	1307.95 (44)	16	32	80	20	NA	1000 (44)
3) Experiments	324 (52)	1050 (48)	16	40 (18)	NA	NA	2000 (48)	8800 (52)
Totals (Mainframe core only)	324 (46)	1732.0 (46)	16	40 (44)	85	15	2000 (46)	8800 (46)
Totals (Worst Case Values)	324 (46)	1732.0 (46)	16	40 (44)	85 (42)	15 (42)	2000 (46)	8800 (46)

NA = Not Available

Table 3-1
(continued)

PHASE HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters / Phases	(K) Ops/Secord (29)		Word Length (Bits) (29)		Instruction Mix (%) (29)		I/O Requirements Bits x 10^3/sec (29)	
	Min	Max	Min	Max	Add	Mult	Min	Max
1) Boost								
a) Powered Ascent	30.16	585.45	16	32	89	11	44.7	330
b) Insertion	28.38	736.77	16	32	88	12	44.7	330
2) Orbital Free Flight	6.38	1264.16	16	32	89	11	350	800
3) Orbital Powered Flight			16	32				
a) Basic OPF	29.86	994.20	16	32	88	12	44.7	330
b) Rendezvous Targeting/ Docking	56.00	1307.95	16	32	89	11	44.7	330
4) Interplanetary Flight (Extremes)	285.454 (35)	1307.95 (35)	16	32	80	20	NA	1000
5) Experiments	324 (52)	1050 (48)	16	40 (18)	NA	NA	2000 (48)	8800 (52)

NA = Not Available

Table 3-2

PHASE HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Parameters / Phases	Instructions (29)		Data (words) (29)				Total Words (29)	
			Mainframe		Bulk			
	Min	Max	Min	Max	Min	Max	Min	Max
1) Boost								
a) Powered Ascent	9663	40831	3987	21952	None	8000	13650	70783
b) Insertion	9857	47799	4058	21352	None	8000	13915	77151
2) Orbital Free Flight	6795	56162	2928	32748	None	87400	9723	176310
3) Orbital Powered Flight								
a) Basic OPF	10154	47799	4168	21352	None	8000	14322	77151
b) Rendezvous Targeting	9597	40107	3960	22724	None	8000	13557	70831
4) Interplanetary Flight (Extremes)	7690 (35)	47799 (35)	4490 (35)	22724 (35)	None (35)	8000 (35)	12180 (35)	78523 (35)
5) Experiments	21450 (49)	97280 (50)	12550 (49)	44600 (50)	22000 (48)	9772000 (18)	56000	9893000

Table 3-2
(continued)

- Boost
- Orbital Free Flight (OFF)
- Orbital Powered Flight (OPF)
- Interplanetary Flight
- Experiments

While Experiments would not normally be considered as an entirely independent mission phase, it is classified as such here in order to prevent the Experiments parameter estimates from obscuring the other phase estimates.

Both the Boost phase and the Orbital Powered Flight phase have been further partitioned into subphases. The Boost subphases are Powered Ascent and Insertion, and the Orbital Powered Flight subphases are Basic OPF and Rendezvous Targeting/Docking. Each of these phases and subphases are discussed, briefly, later in this report.

3.2.1 Phase Hardware Parameters

The Phase Hardware parameters summarized in Table 3-2 are documented in considerably more detail in Tables 3-2a through 3-2f in this subsection. Each defined phase or subphase, except for Experiments, has been documented separately.

The Experiments phase could not be further broken down since insufficient information was available in the reference material reviewed. While substantial information was available defining Experiment types, little data was available defining onboard processor participation requirements in experiment activity.

In order to establish a common baseline for evaluating phase processor requirements, phases were standardized as consisting of eight tasks:

- Guidance
- Navigation
- Control

- Executive
- Display and Communication
- Fault Isolation and Redundancy Management
- Subsystem Processing
- Utilities

Fluctuations in task sizing estimates due to mission phase re-requirements, which resulted in significant variations in the parameter estimates, were isolated by defining multiple tasks with the same standardized name; i.e., Guidance A, Guidance B, etc.

The Interplanetary phase was not broken down into the above defined standardized tasks since the associated reference material did not treat Interplanetary Flight from a task viewpoint. The Interplanetary Flight phase MAP is discussed in detail in Section 3.2.1.4.

3.2.1.1 Boost Phase

Boost is that mission phase during which the vehicle payload is transferred from the launch site to earth orbit. It consists of two principle subphases: Powered Ascent and Insertion. Insertion differs from Powered Ascent only in that the final control parameters, including the exact cutoff time, must be precisely calculated.

Tables 3-2a and 3-2b define the phase hardware parameters for Powered Ascent and Insertion respectively.

3.2.1.2 Orbital Free Flight Phase

Orbital Free Flight (OFF) is that mission phase during which the vehicle is operational in a preachieved orbit. During this phase navigation and attitude control are normally performed, but guidance and propulsion control are minimal.

The Orbital Free Flight phase hardware parameters are defined in Table 3-2c.

3.2.1.3 Orbital Powered Flight Phase

Orbital Powered Flight (OPF) is that mission phase during which the vehicle changes or corrects its orbit. This type of phase would

POWERED ASCENT HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters / Tasks	(K) Ops/Second (27)		Word Length (bits) (26)		Instruction Mix (%) (28)		I/O Requirements Bits x 10^3/sec	
	Min	Max	Min	Max	Add	Mult	Min	Max
Guidance A	1.04	11.00	16	32	80	20	3.8 (36)	NB
Navigation A	.52	19.10	16	32	80	20	3.8 (36)	NB
Control A	16.63	148.95	16	32	80	20	14.8 (37)	NB
Executive	10.20	25.50	N/A	N/A	100	0	0 (38)	0 (38)
Display & Communication	.77	96.00	16	32	100	0	10 (37)	NB
Fault Isolation & Redundancy Management	.07	32.25	16	32	100	0	.08 (39)	NB
Subsystem Processing	.63	212.50	16	32	90	10	11.5 (37)	NB
Utilities	.30	40.15	16	32	80	20	NA	NB
Totals	30.16	585.45	16	32	89.16 (42)	10.84 (42)	44.7	330 (18)

NA = Not Available NB = No Breakdown N/A = Not Applicable

Table 3-2a

POWERED ASCENT HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Parameters / Tasks	Instructions (28)		Data (words) (28)				Total Words (28)	
			Mainframe		Bulk			
	Min	Max	Min	Max	Min	Max	Min	Max
Guidance A	520	879	192	325	None	None	712	1204
Navigation A	1036	3820	382	680	None	None	1418	4500
Control A	3263	5958	1205	3480	None	None	4468	9438
Executive	1020	5844	180	647	None	8000	1200	14491
Display & Communication	765	4800	135	1600	None	None	900	6400
Fault Isolation & Redundancy Management	1497	6450	877	9300	None	None	2374	15750
Subsystem Processing	1262	4250	816	750	None	None	2078	5000
Utilities	300	8830	200	5170	None	None	500	14000
Totals	9663	40831	3987	21952	None	8000	13650	70783

Table 3-2a
(continued)

INSERTION HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters	(K) Ops/Second (27)		Word Length (Bits) (26)		Instruction Mix (%) (28)		I/O Requirements Bits x 10^3/sec	
	Min	Max	Min	Max	Add	Mult	Min	Max
Guidance B	4.02	33.87	16	32	80	20	3.8 (36)	NB
Navigation A	.52	19.10	16	32	80	20	3.8 (36)	NB
Control B	11.87	277.40	16	32	80	20	14.8 (37)	NB
Executive	10.20	25.50	N/A	N/A	100	0	0 (38)	0 (38)
Display & Communication	.77	96.00	16	32	100	0	10 (37)	NB
Fault Isolation & Redundancy Management	.07	32.25	16	32	100	0	.08 (39)	NB
Subsystem Processing	.63	212.50	16	32	90	10	11.5 (37)	NB
Utilities	.30	40.15	16	32	80	20	NA	NB
Totals	28.38	736.77	16	32	88.04 (42)	11.96 (42)	44.7	330 (18)

NA = Not Available NB = No Breakdown N/A = Not Applicable

Table 3-2b

INSERTION HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Parameters / Tasks	Instructions (28)		Data (words) (28)				Total Words (28)	
			Mainframe		Bulk			
	Min	Max	Min	Max	Min	Max	Min	Max
Guidance B	1604	2710	592	1000	None	None	2196	3710
Navigation A	1036	3820	382	680	None	None	1418	4500
Control B	2373	11095	876	2205	None	None	3249	13300
Executive	1020	5844	180	647	None	8000	1200	14491
Display & Communication	765	4800	135	1600	None	None	900	6400
Fault Isolation & Redundancy Management	1497	6450	877	9300	None	None	2374	15750
Subsystem Processing	1262	4250	816	750	None	None	2078	5000
Utilities	300	8830	200	5170	None	None	500	14000
Totals	9857	47799	4058	21352	None	8000	13915	77151

Table 3-2b
(continued)

ORBITAL FREE FLIGHT HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters / Tasks	(K) Ops/Second (27)		Word Length (Bits) (26)		Instruction Mix (%) (28)		I/O Requirements Bits x 10^3/sec	
	Min	Max	Min	Max	Add	Mult	Min	Max
Guidance D	.00	.87	16	32	80	20	NB	NB
Navigation B	.95	175.00	16	32	80	20	NB	NB
Control E	3.76	554.80	16	32	80	20	NB	NB
Executive	.05	58.44	N/A	N/A	100	0	NB	NB
Display & Communication	.77	312.00	16	32	100	0	NB	NB
Fault Isolation & Redundancy Management	.07	32.25	16	32	100	0	NB	NB
Subsystem Processing	.63	42.50	16	32	90	10	NB	NB
Utilities	.15	88.30	16	32	80	20	NB	NB
Totals	6.38	1264.16	16	32	89.0 (42)	11.0 (42)	350 (9)	800 (40)

NB = No Breakdown N/A = Not Applicable

Table 3-2c

ORBITAL FREE FLIGHT HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Parameters / Tasks	Instructions (28)		Data (words) (28)				Total Words (28)	
			Mainframe		Bulk			
	Min	Max	Min	Max	Min	Max	Min	Max
Guidance D	252	433	93	592	None	3800	345	4825
Navigation B	947	3500	349	2368	None	15200	1296	21068
Control E	752	11095	278	2960	None	19000	1030	33055
Executive	1020	5844	180	2368	None	8000	1200	16212
Display & Communication	765	15760	135	9240	None	30000	900	55000
Fault Isolation & Redundancy Management	1497	6450	877	9300	None	11400	2374	27150
Subsystem Processing	1262	4250	816	750	None	None	2078	5000
Utilities	300	8830	200	5170	None	None	500	14000
Totals	6795	56162	2928	32748	None	87400	9723	176310

Table 3-2c
(continued)

normally be utilized by a taxi or tug vehicle in earth orbit or by a space station making major orbital corrections in order to prevent orbital decay. OPF as defined for this MAP consists of two subphases, Basic OPF and Rendezvous Targeting/Docking.

Basic OPF consists of the primary guidance, navigation, and control operations necessary to change orbits in a controlled manner. Rendezvous Targeting/Docking consists of those additional guidance and control operations required to intercept and/or mate with a target vehicle.

The Basic OPF and Rendezvous Targeting/Docking subphase hardware parameters are defined in Tables 3-2d and 3-2e.

3.2.1.4 Interplanetary Flight Phase

Interplanetary Flight is that mission phase during which the vehicle is transferred from planetary orbit about one planet to planetary orbit about another planet.

The reference material reviewed during this study associated with Interplanetary Flight was relatively sketchy. It was not defined in a manner which permitted it to be broken down into the standardized tasks used to describe the previously discussed mission phases. Therefore, the Interplanetary Flight phase hardware parameters defined in Table 3-2f detail Interplanetary Flight subphases rather than Interplanetary Flight tasks.

3.2.2 Software Structural Parameters

The Software Structural Parameters are defined to be those parameters which are mission phase or subphase dependent and which are directly influenced by the mission software structural requirements.

During this study two software structural parameters were defined: Task Iteration Rate and Relative Priority.

The Task Iteration Rate defines the number of task executions, required per second, for each phase. The Relative Priority provides a qualitative measurement of task importance with respect to successful mission phase completion.

Table 3-3 defines the software structural parameters for each task of each phase.

BASIC ORBITAL POWERED FLIGHT HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters / Tasks	(K) Ops/Second (27)		Word Length (Bits) (26)		Instruction Mix (%) (28)		I/O Requirements Bits x 10^3/sec	
	Min	Max	Min	Max	Add	Mult	Min	Max
Guidance B	4.02	33.87	16	32	80	20	3.8 (36)	NB
Navigation A	.52	191.00	16	32	80	20	3.8 (36)	NB
Control C	13.35	194.18	16	32	80	20	14.8 (37)	NB
Executive	10.20	146.10	N/A	N/A	100	0	N/A	N/A
Display & Communication	.77	96.00	16	32	100	0	10 (37)	NB
Fault Isolation & Redundancy Management	.07	32.25	16	32	100	0	.08 (39)	NB
Subsystem Processing	.63	212.50	16	32	90	10	11.5 (37)	NB
Utilities	.30	88.30	16	32	80	20	NA	NB
Totals	29.86	994.20	16	32	88.0 (42)	12.0 (42)	44.7	330 (18)

NB = No Breakdown NA = Not Available N/A = Not Applicable

Table 3-2d

BASIC ORBITAL POWERED FLIGHT HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Parameters / Tasks	Instructions (28)		Data (words) (28) Mainframe		Data (words) (28) Bulk		Total Words (28)	
	Min	Max	Min	Max	Min	Max	Min	Max
Guidance B	1604	2710	592	1000	None	None	2196	3710
Navigation A	1036	3820	382	680	None	None	1418	4500
Control C	2670	11095	986	2205	None	None	3656	13300
Executive	1020	5844	180	647	None	8000	1200	14491
Display & Communication	765	4800	135	1600	None	None	900	6400
Fault Isolation & Redundancy Management	1497	6450	877	9300	None	None	2374	15750
Subsystem Processing	1262	4250	816	750	None	None	2078	5000
Utilities	300	8830	200	5170	None	None	500	14000
Totals	10154	47799	4168	21352	None	8000	14322	77151

Table 3-2d

RENDEZVOUS TARGETING/DOCKING HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters / Tasks	(K) Ops/Second (27)		Word Length (Bits) (26)		Instruction Mix (%) (28)		I/O Requirements Bits x 10^3/sec	
	Min	Max	Min	Max	Add	Mult	Min	Max
Guidance C	.03	3.00	16	32	80	20	3.8 (36)	NB
Navigation B	.95	175.00	16	32	80	20	3.8 (36)	NB
Control D	43.05	554.80	16	32	80	20	14.8 (37)	NB
Executive	10.20	146.10	N/A	N/A	100	0	0 (38)	0 (38)
Display & Communication	.77	96.00	16	32	100	0	10 (37)	NB
Fault Isolation & Redundancy Management	.07	32.25	16	32	100	0	.08 (39)	NB
Subsystem Processing	.63	212.50	16	32	90	10	11.5 (37)	NB
Utilities	.30	88.30	16	32	80	20	NA	NB
Totals	56.00	1307.95	16	32	89.13 (42)	10.87 (42)	44.7	330 (18)

NA = Not Available NB = No Breakdown N/A = Not Available

Table 3-2e

RENDEZVOUS TARGETING/DOCKING HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Tasks \ Parameters	Instructions (28)		Data (words) (28)				Total Words (28)	
			Mainframe		Bulk			
	Min	Max	Min	Max	Min	Max	Min	Max
Guidance C	3554	6000	1310	3500	None	None	4864	9500
Navigation B	947	3500	349	1600	None	None	1296	5100
Control D	252	433	93	157	None	None	345	590
Executive	1020	5844	180	647	None	8000	1200	14491
Display & Communication	765	4800	135	1600	None	None	900	6400
Fault Isolation & Redundancy Management	1497	6450	877	9300	None	None	2374	15750
Subsystem Processing	1262	4250	816	750	None	None	2078	5000
Utilities	300	8830	200	5170	None	None	500	14000
Totals	9597	40107	3960	22724	None	8000	13557	70831

Table 3-2e
(continued)

INTERPLANETARY FLIGHT HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters / Subphases	(K) Ops/Second		Word Length (Bits)		Instruction Mix (%) (25)		I/O Requirements Bits x 10^3/sec	
	Min	Max	Min	Max	Add	Mult	Min	Max
Transplanetary Injection	301.254 (4)	736.77 (30)	16	32	80	20	NB	NB
Transplanetary Coast	314.860 (4)	1048.16 (31)	16	32	80	20	NB	NB
Trajectory Correction	379.454 (4)	*1307.95 (32)	16	32	80	20	NB	NB
Spin-Up	308.550 (4)	NE	16	32	80	20	NB	NB
Spin-Cruise	318.660 (4)	1048.16 (31)	16	32	80	20	NB	NB
Despin	308.550 (4)	NE	16	32	80	20	NB	NB
Planetary Approach Correction	379.454 (4)	1307.95 (32)	16	32	80	20	NB	NB
Aero Braking	287.550 (4)	585.45 (33)	16	32	80	20	NB	NB
Planetary Orbital Injection	*285.454 (4)	736.77 (30)	16	32	80	20	NB	NB
Planetary Orbital	1048.16 (31)	*1449.402 (4)	16	32	80	20	NB	NB
*Interplanetary Extremes	285.454	1449.402	16	32	80	20	NA	1000 (41)

NE = No Equivalents NB = No Breakdown

Table 3-2f

INTERPLANETARY FLIGHT HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Parameters / Tasks/Subphases	Instructions		Data (words) Mainframe		Data (words) Bulk		Total Words	
	Min (14)	Max	Min (14)	Max	Min (4)	Max	Min (4)	Max
Transplanetary Injection	8129	47799* (30)	4740	21352 (30)	None	8000* (30)	12869	77151 (30)
Transplanetary Coast	11286	45202 (31)	6600	21429 (31)	None	8000 (31)	17886	74631 (31)
Trajectory Correction	8074	40107 (32)	4725	22724* (32)	None	8000 (32)	12799	70831 (32)
Spin-Up	7690*	NE	4490*	NE	None	NE	12180	NE
Spin Cruise	11656	45202 (31)	6820	21429 (31)	None	8000 (31)	18476	74631 (31)
Despin	7690	NE	4490	NE	None	NE	12180	NE
Planetary Approach Correction	8779	40107 (32)	5140	22724 (32)	None	8000 (32)	13919	70831 (32)
Aero Braking	9160	40831 (33)	5360	21952 (33)	None	8000 (33)	14520	70783 (33)
Planetary Orbital Injection	7859	47799 (30)	4590	21352 (30)	None	8000 (30)	12449	77151 (30)
Planetary Orbital	15513	45202 (31)	9120	21429 (31)	None	8000 (31)	24633	74631 (31)
*Interplanetary Extremes	7690	47799	4490	22724	None	8000	12180	78523 (34)

NE = No Equivalent

Table 3-2f
(continued)

BOOST-POWERED ASCENT

SOFTWARE STRUCTURAL PARAMETERS

Tasks \ Parameters	Iteration Rate (executions/sec)		Relative Priority(25)
	Min	Max	1 = High 2 = Medium 3 = Low
Guidance A	4 (20)	25 (21)	1
Navigation A	1 (21)	10 (20)	1
Control A	10 (22)	50 (22)	1
Executive	20 (20)	50 (21)	1
Display & Communication	2 (22)	40 (20)	3
Fault Isolation & Redundancy Management	.1 (19)	10 (20)	2
Subsystem Processing	1 (21)	100 (22)	2
Utilities	2 (23)	10 (22)	2

Table 3-3

BOOST - INSERTION

SOFTWARE STRUCTURAL PARAMETERS

Tasks \ Parameters	Iteration Rate (executions/sec)		Relative Priority (25)
	Min	Max	1 = High 2 = Med. 3 = Low
Guidance B	5 (20)	25 (21)	1
Navigation A	1 (21)	10 (20)	1
Control B	10 (22)	50 (22)	1
Executive	20 (20)	50 (21)	1
Display & Communication	2 (22)	40 (20)	3
Fault Isolation & Redundancy Management	.1 (19)	10 (20)	2
Subsystem Processing	1 (21)	100 (22)	2
Utilities	2 (23)	10 (22)	2

Table 3-3
(continued)

ORBITAL FREE FLIGHT

SOFTWARE STRUCTURAL PARAMETERS

Tasks \ Parameters	Iteration Rate (executions/sec) Min	Max	Relative Priority (25) 1 = High 2 = Med. 3 = Low
Guidance D	.0166 (23)	4 (20)	3
Navigation B	2 (20)	100 (9)	2
Control E	10 (9)	100 (20)	1
Executive	.1 (19)	20 (20)	1
Display & Communication	2 (22)	40 (20)	2
Fault Isolation & Redundancy Management	.1 (19)	10 (20)	2
Subsystem Processing	1.0 (19)	20 (9)	2
Utilities	1 (19)	20 (9)	2

Table 3-3
(continued)

ORBITAL POWERED FLIGHT - BASIC

SOFTWARE STRUCTURAL PARAMETERS

Tasks \ Parameters	Iteration Rate (executions/sec) Min	Max	Relative Priority (25) 1 = High 2 = Med. 3 = Low
Guidance B	5 (20)	25 (21)	1
Navigation A	1 (21)	100 (9)	1
Control C	10 (22)	35 (20)	1
Executive	20 (20)	50 (21)	1
Display & Communication	2 (22)	40 (20)	3
Fault Isolation & Redundancy Management	.1 (19)	10 (20)	2
Subsystem Processing	1 (21)	100 (22)	2
Utilities	2 (23)	20 (9)	2

Table 3-3
(continued)

ORBITAL POWERED FLIGHT - RENDEZVOUS TARGETING/DOCKING

SOFTWARE STRUCTURAL PARAMETERS

Tasks \ Parameters	Iteration Rate (executions/sec) Min	Max	Relative Priority (25) 1 = High 2 = Med. 3 = Low
Guidance C	.0166 (23)	1.0 (20)	1
Navigation B	2 (20)	100 (9)	1
Control D	25 (20)	100 (9)	1
Executive	20 (20)	50 (21)	1
Display & Communication	2 (22)	40 (20)	3
Fault Isolation & Redundancy Management	.1 (19)	10 (20)	2
Subsystem Processing	1 (21)	100 (22)	2
Utilities	2 (23)	20 (9)	2

Table 3-3
(continued)

3.3 Task Profiles

The Task Profiles consist of the Task Hardware parameters and the Task Timing and Criticality parameters defined in this section.

The Task Hardware parameters are defined to be those parameters which are task dependent and which have a direct influence on the processor hardware characteristics; i.e., Word Length, Memory Size, Execution Speed, etc. These parameters are presented in Section 3.3.1.

The Task Timing and Criticality parameters are defined to be those parameters which for each task define its operating criteria and constraints. These parameters are presented in Section 3.3.2.

Sixteen (16) distinct tasks were identified during this study. Each of these tasks, along with its associated parameter values, is listed in the tables of this section.

3.3.1 Task Hardware Parameters

Table 3-4 presents the Task Hardware parameters for each of the sixteen (16) principal tasks identified during this study. In general, the range of variation, as defined in the reference material, is presented for each task parameter.

3.3.2 Task Timing and Criticality Requirements

Table 3-5 presents the Task Timing and Criticality Requirements for each of the sixteen (16) defined tasks. Most of these parameters were established by M&S Computing as indicated by the footnote.

The Criticality parameters, Restart Timing Sensitivity, Interruptability, Failure Criticality, and Memory Duplicates desired are highly subjective and no attempt has been made in this MAP to provide anything other than a gross qualitative indication of criticality.

The Failure Criticality parameters are of special interest to the ARMMS application and are therefore defined here a little more explicitly than on the footnotes of Table 3-5.

D The failure must be detected and output from the failing task prevented. Correct output must be provided within the time frame indicated by the Restart Timing Sensitivity parameter.

TASK HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Tasks \ Parameters		(K) Ops/Iteration (24)		Word Length (Bits)		Instruction Mix (%) (22)		I/O Requirements Bits x 10^3/iteration	
Name	Description	Min	Max	Min (5)	Max(1)	Add	Mult	Min	Max
Guidance A	Ascent & Separation	.260	.440	15	32	80	20	NB	NB
Guidance B	Insertion & OPF	.804	1.355	15	32	80	20	NB	NB
Guidance C	Rendezvous Targeting	1.777	3.000	15	32	80	20	NB	NB
Guidance D	Orbital Free Flight (OFF)	.126	.217	15	32	80	20	NB	NB
Navigation A	Boost & OPF	.518	1.910	15	32	80	20	NB	NB
Navigation B	Rendezvous Tgt & OFF	.474	1.750	15	32	80	20	NB	NB
Control A	Ascent & Separation	1.632	2.979	15	32	80	20	NB	NB
Control B	Insertion	1.187	5.548	15	32	80	20	NB	NB
Control C	OPF	1.335	5.548	15	32	80	20	NB	NB
Control D	Rendezvous Targeting	1.722	5.548	15	32	80	20	NB	NB
Control E	OFF	.376	5.548	15	32	80	20	NB	NB

NB = No Breakdown Available

Table 3-4

TASK HARDWARE PARAMETERS

PROCESSOR CAPABILITY

Parameters / Tasks	(K) Ops/Iteration(24)		Word Length (Bits)		Instruction Mix (%) (22)		I/O Requirements Bits x 10^3/iteration	
	Min	Max	Min	Max	Add	Mult	Min	Max
Executive	.510	2.922	N/A	N/A	100	0	0 (38)	0 (38)
Display & Communication	.383	7.800	15	32	100	0	NB	NB
Fault Isolation & Redundancy Management	.749	3.225	15	32	100	0	NB	NB
Subsystem Processing	.631	2.125	15	32	90	10	NB	NB
Utilities	.150 (19)	4.415	15	32	80 (25)	20 (25)	NB	NB

NB = No Breakdown Available N/A = Not Applicable

Table 3-4
(continued)

TASK HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Tasks / Parameters		Instructions		Data (words)				Total Words	
				Mainframe		Bulk			
Name	Description	Min	Max	Min	Max	Min	Max	Min	Max
Guidance A	Ascent & Separation	520 (2)	879 (8)	192 (3)	325 (8)	None	None	712	1204
Guidance B	Insertion & OPF	1604 (2)	2710 (8)	592 (3)	1000 (8)	None	None	2196	3710
Guidance C	Rendezvous Targeting	3554 (2)	6000 (7)	1310 (3)	3500 (7)	None	None	4864	9500 (6)
Guidance D	OFF	252 (2)	433 (8)	93 (3)	592 (52)	None	3800 (52)	345	4825
Navigation A	Boost & OPF	1036 (2)	3820 (11)	382 (3)	680 (11)	None	None	1418	4500 (6)
Navigation B	Rendezvous Tgt & OFF	947 (2)	3500 (10)	349 (3)	2368 (52)	None	15200 (52)	1296	21068 (9)
Control A	Ascent & Separation	3263 (2)	5958 (14)	1205 (3)	3480 (14)	None	None	4468	9438 (12)
Control B	Insertion	2373 (2)	11095 (14)	876 (3)	2205 (14)	None	None	3249	13300 (13)
Control C	OPF	2670 (2)	11095 (14)	986 (3)	2205 (14)	None	None	3656	13300 (13)
Control D	Rendezvous Targeting	3444 (2)	11095 (14)	1272 (3)	2205 (14)	None	None	4716	13300 (13)
Control E	OFF	752 (2)	11095 (14)	278 (3)	2960 (14)	None	19000 (52)	1030	33055 (13)

Table 3-4
(continued)

TASK HARDWARE PARAMETERS

MEMORY REQUIREMENTS

Tasks \ Parameters	Instructions		Data (words) Mainframe		Data (words) Bulk		Total Words	
	Min	Max	Min	Max	Min	Max	Min	Max
Executive	1020 (10)	5844 (15)	180 (10)	2360 (52)	None	8000 (18)	1200 (10)	16212
Display & Communication	765 (10)	15760 (52)	135 (10)	9240 (52)	None	3000 (52)	900 (10)	55000
Fault Isolation & Redundancy Management	1497 (14)	6450 (17)	877 (14)	9300 (17)	None	114001 (52)	2374 (12)	27150
Subsystem Processing	1262 (19)	4250 (10)	816 (19)	750 (10)	None	None	2078 (19)	5000 (10)
Utilities	300 (19)	8830 (14)	200 (19)	5170 (14)	None	None	500 (19)	14000 (6)

Table 3-4
(continued)

TASK TIMING AND CRITICALITY PARAMETERS

Tasks \ Parameters	Restart Timing Sensitivity*	Interrupt Ability **	Failure Criticality ***	Memory Duplicates Desirable	Task Duration (ms) (assumes 2μs add)**** Min per Pass	Max per Pass
Guidance A	M (25)	CI (25)	D (25)	Yes (25)	.832	1.406
Guidance B	M (25)	CI (25)	D (25)	Yes (25)	2.566	4.336
Guidance C	M (25)	CI (25)	D (25)	Yes (25)	5.686	9.600
Guidance D	M (25)	CI (25)	D (25)	Yes (25)	.403	.693
Navigation A	M (25)	CI (25)	D (25)	Yes (25)	1.658	6.112
Navigation B	M (25)	CI (25)	D (25)	Yes (25)	1.512	5.600

* H = High (sensitivity impact within 1 minor cycle), M = medium sensitivity (within 1 major cycle), L = low sensitivity (greater than 1 major cycle)

** I = Interruptable, NI = Not interruptable, CI = Conditionally interruptable

*** M = Failures must be masked, D = Failures must be identified, N = None (no failure criteria)

**** Assumes 50% of instructions executed/iteration with 2 μs add time and instruction mix defined
8 μs mult time

Table 3-5

TASK TIMING AND CRITICALITY PARAMETERS

Tasks \ Parameters	Restart Timing Sensitivity*	Interrupt Ability **	Failure Criticality ***	Memory Duplicates Desired	Task Duration (ms) (2 μs add, 8 μs mult) Min per Pass	Max per Pass
Control A	H (25)	NI (25)	D (25)	Yes (25)	5.221	9.533
Control B	H (25)	NI (25)	D (25)	Yes (25)	3.797	17.752
Control C	H (25)	NI (25)	D (25)	Yes (25)	4.272	17.752
Control D	H (25)	NI (25)	D (25)	Yes (25)	5.510	17.752
Control E	H (25)	NI (25)	D (25)	Yes (25)	1.203	17.752
Executive	H (25)	I (25)	D (25)	Partial (25)	1.020	5.844
Display & Communication	L (25)	I (25)	N (25)	No (25)	.765	4.800
Fault Isolation & Redundancy Management	L (25)	CI (25)	D (25)	Partial (25)	1.497	6.450
Subsystem Processing	M (25)	CI (25)	D (25)	Yes (25)	1.641	5.525
Utilities	H (25)	I (25)	D (25)	Yes (25)	.480	14.128

* H = High (sensitivity impact within 1 minor cycle), M = Medium sensitivity (within 1 major cycle), L = low sensitivity (greater than 1 major cycle)

** I = Interruptable, NI = Not Interruptable, CI = Conditionally Interruptable

*** M = Failures must be masked, D = Failures must be identified, N = None (no failure criteria)

Table 3-5
(continued)

N Output from this task under failure conditions need not be prevented, even though failure must be detected.

,.4 Reviewed Mission Summary

This paragraph documents in detail the parameters associated with the key missions studied. Most of the tabulated data in Section 3 was derived from the following references:

SH-4 Participation in Phase B Space Shuttle Program Definition; March 2, 1971, IBM.

M-1 Study of Spaceborne Multiprocessing; September, 1968, Autonetics.

SS-5 Reconfigurable G&C Computer Study for Space Station Use; January 31, 1971, Autonetics.

SH-2 Data Management System Control Study - Technical Design Notes 1-21; May to September, 1971, TRW.

SH-8 Phase B Shuttle Review; September, 1970, IBM.

AS-4 Saturn LVDC Statistics; February 8, 1971, MSFC.

The other references in Appendix 2 were used in deriving the MAP, but the listed references provided the bulk of the data. Tables 3-6 through 3-11 in this paragraph depict the exact data gathered for each key reference. All blank entries represent data not available in this reference or not applicable.

These tables are oriented around the standardized tasks defined during the evaluation. This approach for presentation of the backup material was chosen as an alternate to presenting the source material in the format as used by the originators, to simplify the data presented and to establish a baseline format conducive to cross comparison. No attempt has been made in this section to interpret or explain the source data. Readers interested in this type of information should reference the source material.

SOURCE SH-4

OVERALL PHASE HARDWARE PARAMETERS

Parameters / Phases	Processor Capability		Memory Requirements				I/O Requirements	
	Ops/Sec (K)	Word Length (Bits)	Instructions (words)	Data: Mainframe (words)	Data: Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)
1) Boost								
a) Powered Ascent	156.36	32				24342		
b) Insertion	168.10	32				24536		
2) Orbital Free Flight	148.15	32				21474		
3) Orbital Powered Flight								
a) Basic Orbital Pwr Flt	165.70	32				24833		
b) Rendezvous Tgt	152.25	32				27468		

Table 3-6

SOURCE SH-4

POWERED ASCENT HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability		Memory Requirements				I/O Requirements	
			Instructions (words)	Data		Total (words)		
	Ops/Sec (K)	Word Length (Bits)		Mainframe (words)	Bulk (words)		Input (bytes/sec)	Output (bytes/sec)
Guidance A	1.06	32	520			520		
Navigation A	4.9	32	1036			1036		
Control A	41.2	32	3263			3263		
Executive	21.9	32	2420			2420		
Displays & Data Mgt	58.2	32	2980			2980		
Fault Isolation & Red. Mgt	25.1	32	4082			4082		
Subsystem Processing	4.0	32	2100			2100		
System Tables				7941		7941		
Total	156.36	32	16406	7941		24342		

Table 3-6a

SOURCE SH-4

INSERTION HARDWARE PARAMETERS

Tasks \ Parameters	Processor Capability		Memory Requirements				I/O Requirements	
	Ops/Sec (K)	Word Length (Bits)	Instructions (words)	Data: Mainframe (words)	Data: Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)
Guidance B	5.0	32	1604			1604		
Navigation A	4.9	32	1036			1036		
Control B	48.9	32	2373			2373		
Executive	21.9	32	2420			2420		
Displays & Data Mgt	58.2	32	2980			2980		
Fault Isolation & Red. Mgt	25.1	32	4082			4082		
Subsystem Processing	4.1	32	2100			2100		
System Tables				7941		7941		
Totals	168.1	32	16595	7941		24536		

Table 3-6b

SOURCE SH-4

ORBITAL FREE FLIGHT HARDWARE PARAMETERS

Tasks \ Parameters	Processor Capability		Memory Requirements				I/O Requirements	
	Ops/Sec (K)	Word Length (Bits)	Instructions (words)	Data: Mainframe (words)	Data: Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)
Guidance D	.53	32	252			252		
Navigation B	.82	32	947			947		
Control E	37.6	32	752			752		
Executive	21.9	32	2420			2420		
Displays & Data Mgt	58.2	32	2980			2980		
Fault Isolation & Red. Mgt	25.1	32	4082			4082		
Subsystem Processing	4.0	32	2100			2100		
System Tables				7941		7941		
Totals	148.15	32	3533	7941		21474		

Table 3-6c

SOURCE SH-4

BASIC ORBITAL POWERED FLIGHT HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability		Memory Requirements				I/O Requirements	
			Instructions (words)	Data		Total (words)		
	Ops/Sec (K)	Word Length (Bits)		Mainframe (words)	Bulk (words)		Input (bytes/sec)	Output (bytes/sec)
Guidance B	5.0	32	1604			1604		
Navigation A	4.9	32	1036			1036		
Control C	46.6	32	2670			2670		
Executive	21.9	32	2420			2420		
Displays & Data Mgt	58.2	32	2980			2980		
Fault Isolation & Red. Mgt	25.1	32	4082			4082		
Subsystem Processing	4.0	32	2100			2100		
System Tables				7941		7941		
Totals	165.70	32	16892	7941		24833		

Table 3-6d

SOURCE SH-4

RENDEZVOUS TARGETING HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability		Memory Requirements				I/O Requirements	
			Instructions	Data		Total		
	Ops/Sec (K)	Word Length (Bits)	(words)	Mainframe (words)	Bulk (words)	(words)	Input (bytes/sec)	Output (bytes/sec)
Guidance C	1.03	32	3554			3554		
Navigation B	.82	32	947			947		
Control D	41.2	32	3444			3444		
Executive	21.9	32	2420			2420		
Displays & Data Mgt	58.2	32	2980			2980		
Fault Isolation & Red. Mgt	25.1	32	4082			4082		
Subsystem Processing	4.0	32	2100			2100		
System Tables				7941		7941		
Totals/Max Ranges	152.25	32	19527	7941		27468		

Table 3-6e

SOURCE SH-4

TASK HARDWARE PARAMETERS

Tasks \ Parameters		Processor Capability		Memory Requirements				I/O Requirements	
				Instructions (words)	Data		Total (words)		
Name	Description	Ops/Sec (K)	Word Length (Bits)		Mainframe (words)	Bulk (words)		Input (bytes/sec)	Output (bytes/sec)
Guidance A	Ascent & Separation	1.06	32				520		
Guidance B	Insertion & OPF	5.0	32				1604		
Guidance C	Rendezvous Targeting	1.03	32				3554		
Guidance D	OFF	.53	32				252		
Navigation A	Boost & OPF	4.9	32				1036		
Navigation B	Rendezvous Tgt & OFF	.82	32				947		
Control A	Ascent & Separation	41.2	32				3263		
Control B	Insertion	48.9	32				2373		
Control C	OPF	46.6	32				2670		
Control D	Rendezvous Targeting	41.2	32				3444		
Control E	OFF	37.6	32				752		
Executive		21.9	32				2420		
Displays & Data Mgt		58.2	32				2980		

Table 3-6f

SOURCE SH-4

TASK HARDWARE PARAMETERS

Tasks: Name	Tasks: Description	Processor Capability: Ops/Sec (K)	Processor Capability: Word Length (Bits)	Memory Requirements: Instructions (words)	Memory Requirements: Data: Mainframe (words)	Memory Requirements: Data: Bulk (words)	Memory Requirements: Total (words)	I/O Requirements: Input (bytes/sec)	I/O Requirements: Output (bytes/sec)
Fault Isol. Red. Mgt		25.1	32				4082		
Subsystem Processing		4.0	32				2100		
System Tables	Data				7941		7941		

Table 3-6f
(continued)

SOURCE M-1

INTERPLANETARY FLIGHT HARDWARE PARAMETERS

Parameters / Subphases	Processor Capability		Memory Requirements				I/O Requirements	
	Ops/Sec (K)	Word Length (Bits)	Instructions (words)	Data: Mainframe (words)	Data: Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)
INTP								
1. Trans Plan Inj	57554					6161		
2. Trans Plan Coast	314860					17886		
3. Trajectory Corr	379454					12799		
4. Spin-Up	308550					12180		
5. Spin Cruise	318660					18476		
6. Despin	308550					12180		
7. Plan Appr Corr	379454					13919		
8. Aero Braking	287550					14520		
9. Plan Orbit Inj	285454					12449		
10. Plan Orbit	1449402					24633		
11. Trans Earth Inj	301254					12869		
Totals	4390742					158072		

Table 3-7

SOURCE SS-5

TASK HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability: Ops/Sec (K)	Processor Capability: Word Length (Bits)	Memory Requirements: Instructions (words)	Memory Requirements: Data: Mainframe (words)	Memory Requirements: Data: Bulk (words)	Memory Requirements: Total (words)	I/O Requirements: Input (bytes/sec)	I/O Requirements: Output (bytes/sec)	Software Struct. Parameters: Iteration Rate (per sec)	Software Struct. Parameters: Relative Priority
1) Guidance										
2) Navigation	170	32	3500	1600		5100			100	
3) Control										
a) Maneuver Determination	60.8	32	935	165		1100			20/200	
b) CMG Control	64	32	2510	590		3100			20	
c) RCS Control	200	32	4650	1050		5700			10/200	
d) Attitude Determination	320	32	3000	400		3400			100	
4) Executive	13.2	32	1020	180		1200				
5) Display & Communication	12	32	765	135		900				
6) Fault Isolation & Red. Mgt.	12.2	32	1020	180		1200				

Table 3-8

SOURCE SS-5

TASK HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability		Memory Requirements				I/O Requirements		Software Struct. Parameters	
	Ops/Sec (K)	Word Length (Bits)	Instructions (words)	Data: Mainframe (words)	Data: Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)	Iteration Rate (per sec)	Relative Priority
7) Subsystem Processing										
a) Exp. Module Update		32	3400	600		4000				
b) Taxi Module Align		32	850	150		1000			20	
8) Utilities										
a) Rendezvous			1800	1200		3000			1	
b) Docking	41.8		1700	500		2200			20	
c) Balance Control	18.2		4200	2300		6500			20	
d) Gen. Utilities	N.A.		840	360		1200				
Totals	917.2		30190	9410		39600				

Table 3-8
(continued)

SOURCE SH-2

OVERALL PHASE TASK HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability		Memory Requirements				I/O Requirements		Iteration Rate
	Ops/Sec (K)	Word Length (Bits)	Instructions (words)	Data: Mainframe (words)	Data: Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)	
1) Boost	250000	32				22000	3570	1065	
2) Orbital Powered Flight	250000	32				2300	3816	1794	
3) Orbital Free Flight	250000	32				23000	630	714	

Table 3-9

SOURCE SH-2

OVERALL PHASE TASK HARDWARE PARAMETERS

Tasks \ Parameters	Processor Capability		Memory Requirements				I/O Requirements		Iteration Rate
	Exec. Time 10^{-3} sec	Word Length (Bits)	Instructions (words)	Data: Mainframe (words)	Data: Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)	
Guidance	5.8	32				1200			1
Attitude	.2								25
Navigation	2.5	32				350			1
Control	3.2	32				1350			50
IMU & Rate Gyro	2.2					275			
Main Engines	.8					1750			
Executive	2.6	32				700			50
Data Bus Control						1050			
Display & Data Mgt		32				5800			
Keyboard	2.0								2
Controls & Disp.	6.0								12
Crew	1.0								12
Checkout & Fault Isolation						4000			
Self Test	16	32				1000			
Subsystems									
Sensor Process.	4.4	32				925			1
Vehicle Subsys.	27					2100			1
Communications	2					200			1
Totals						20700			

Table 3-9
(continued)

SOURCE SH-8

TASK HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability			Memory Requirement				Software Struct. Parameters	
	Ops/Sec (K)	Word Length (Bits)	Inst. Mix Add/Mult (%)	Instructions/ words	Data: Mainframe (words/bits)	Data: Bulk (words)	Total (words)	Iteration Rate (per sec)	Relative Priority
1) Guidance	12.0	16/32	80/20	4100/2563	740/32		3303	.2/20	
2) Navigation	9.25	16/32	80/20	4000/3125	1800/32		4925	.01/5110	
3) Control	118.28	16/32	80/20	3300/2063	1000/16		2563	10/50	
4) Executive					4240/16		2120		
a) Control	21.2	16/32	100/0	3584/2638			2638	600	
b) I/O Control	50.0	16/32	100/0	800/600			600	400	
c) Interrupt Processing	27.9	16/32	100/0	360/270			270	370	
d) Flt Control	15.0	16/32	100/0	1100/825	250/16		950	150	
5) Display & Communication	8.9	16/32	100/0	4800/4350	1600/16		5150	2/20	
6) Fault Isolation & Redundancy Mgt									
a) LOFI (checkout & fault isolation	15.5	16/32	100/0	3950/3400	2300/16		4550	80/100	
b) Recovery Mgt	N.A.	16/32		2500/1563	7000/16		5063	N.A.	

Table 3-10

SOURCE SH-8

TASK HARDWARE PARAMETERS

Parameters / Tasks	Processor Capability			Memory Requirement				Software Struct. Parameters	
	Ops/Sec (K)	Word Length (Bits)	Inst. Mix Add/Mult (%)	Instructions/ words	Data Mainframe (words/bits)	Data Bulk (words)	Total (words)	Iteration Rate (per sec)	Relative Priority
7) Subsystem Processing									
a) IOCS	20.0	16/32	100/0	600/450	60/16		480	100	
b) IMU	45.0	16/32	80/20	450/282	40/16		302	100	
8) Utilities									
a) Sequencing	2.0	16/32	100/0	1100/688	1200/32		1888	10	
Totals	345.03			31644/22817	20230/11985		34802		

Table 3-10
(continued)

SOURCE AS-4

TASK HARDWARE PARAMETERS

Parameters	Processor Capability		Memory Requirements				I/O Requirements	
	Ops/Sec (K)	Word Length (Bits)	Instructions (words)	Data Mainframe (words)	Data Bulk (words)	Total (words)	Input (bytes/sec)	Output (bytes/sec)
1) Guidance						9.5K (15%)		
2) Navigation						4.5K (7%)		
3) Control						3K (5%)		
4) Executive						2.5K (4%)		
5) Display & Communication						8.5K (13%)		
6) Fault Isolation & Red. Mgt						9K (14%)		
7) Subsystem Processing						2.5K (4%)		
8) Utilities						14K (22%)		
Prelaunch Checkout & Inflight Spare						10.5K (16%)		
Total						64		

Table 3-11

Table 3-12 and 3-13 present the phase timings for a Mars mission and for a Jupiter/Saturn/Pluto Flyby mission. While the phase timing data was not required for the development of the MAP presented in Section 3, it may be of possible interest. The information presented in Table 3-12 was extracted from Source M-1 (Reference Appendix 2) and the data for Table 3-13 was extracted from Source M-4.

MARS MISSION

Mission Phase	Time (Hours)
1 - ATM Ascent	.2
2 - Earth Orbiter Coast	8.0
3 - Trans Mars Inj.	.2
4 - Trans Mars Coast	119.8
5 - Traj. Corr	.2
6 - Spin Up	1.2
7 - Spin Cruise	2759.8
8 - De Spin	1.2
9 - Mars Appr. Corr	.2
10 - Aero Braking	.4
11 - Mars Orbit Inj.	.6
12 - Mars Orbital Coast	974.3
13 - Trans Earth Inj.	.2
14 - Trans Earth Coast	119.8
15 - Traj. Corr	.2
16 - Spin Up	1.2
17 - Spin Cruise	6119.8
18 - De Spin	1.2
19 - Earth Appr. Corr	.2
20 - Earth Re-Entry	.2

Table 3-12

JUPITER/SATURN/PLUTO FLYBY MISSION

Mission Phase	Time (Hours)
1 - Launch	.8
2 - Attitude Stabilization	90.0
3 - Earth Jupiter Cruise	10,000.0
4 - Near Earth TCM	8.0
5 - Data Dump Mode	4,000.0
6 - Jupiter Approach Guidance	800.0
7 - Pre-Jupiter TCM	8.0
8 - Jupiter Far Encounter	200.0
9 - Jupiter Near Encounter	175.0
10 - Jupiter Playback	10.0
11 - Post-Jupiter TCM	8.0
12 - Jupiter-Saturn Cruise	10,000.0
13 - Saturn Approach Guidance	90.0
14 - Pre-Saturn TCM	8.0
15 - Saturn Far Encounter	200.0
16 - Saturn Near Encounter	175.0
17 - Saturn Playback	90.0
18 - Post Saturn TCM	8.0
19 - Saturn Pluto Cruise	40,000.0
20 - Pluto Approach Guidance	1,000.0

Table 3-13

JUPITER/SATURN/PLUTO FLYBY MISSION
(continued)

Mission Phase	Time (Hours)
21 - Pre-Pluto TCM	8.0
22 - Pluto Far Encounter	200.0
23 - Pluto Near Encounter	175.0
24 - Pluto Playback	300.0

Table 3-13
(continued)

4. SATURN V - BOOST PROFILE

Although a great deal of information can be derived from the MAP described in Section 3, there are several details that cannot be derived from the functional level used for the MAP.

To illustrate the additional details it was necessary to analyze an actual, operational, typical spaceborne application. This application selected was the Saturn V Flight Program. Specifically, the application modules involved in the boost phase were analyzed, for the simple reason that this phase clearly dominates the other mission phases on Saturn V.

Table 4-1 presents the various sizing, activation etc. parameters of the individual tasks. Note that "tasks" within this context reflect a more detailed breakdown than the "tasks" described in Section 3.

The Events Sequence Time Line underlying the execution of the tasks is depicted in Table 4-2.

Note that the tasks that could be performed as part of a Control Executive are not reflected in this profile.

SATURN V BOOST TASK PROFILE

Task Name/ID	Execution Time (add equiv) Nominal	Execution Time (add equiv) Maximum	Memory Req. (# of instr.)	Synchr/ Asynchr	Iteration Rate	Activate Upon	Deactivate Upon	Preceding Task	Completion Deadline	Parameters Input	Parameters Output	Telem. (words per equiv)	Inter. Inhb. (add equiv.)
Accelerometer Processing/AP	314	343	220	S	Major Loop	Phase Start	Phase End	AR	Major Loop	21	5	3	None
Accelerometer Read/AR	573	576	215	S	Major Loop	Phase Start	Phase End	Start Major Loop	Major Loop	13	23	7	41
CHI Computations/CC	286	286	53	S	Major Loop	Reference IG	Reference IG	IG	Major Loop	18	2	3	None
Cutoff Logic (CO)	195	195	115	S	25/sec	Reference HS	Reference HS	ML	Major Loop	9	4	0	Duration of Execution
Disagreement Bit Processing (DG)	31	90	181	A	--	Detection of gimbal disag. bit by ML	Single Execution	ML	Reference ML	7	6	1	Duration of Execution
Discrete Input Processor #10 (DP)	13	13	#10, #15 40	A	--	S-II/S-IVB Separation	Single Execution	--	Major Loop	#10, #15 0	#10, #15 0	#10, #15 0	None
Discrete Input Processor #11 (DP)	56	56	#11, #14, #19 137	A	--	S-IC Inboard Engine out	Single Execution	--	Major Loop	#11, #14, #19 6	#11, #14, #19 8	#11, #14, #19 1	None

Table 4-1

SATURN V BOOST TASK PROFILE

Task Name/ID	Execution Time (add equiv)		Memory Req. (# of instr.)	Synchr/ Asynchr	Iteration Rate	Activate Upon	Deactivate Upon	Preceding Task	Completion Deadline	Parameters		Telem. (words per equiv)	Inter. Inhb. (add equiv.)
	Nominal	Maximum								Input	Output		
Discrete Input Processor #13 (DP)	100	100	#13, #21 136	A	--	S-II Inboard Engine Out	Single Execution	--	Major Loop	#13, #21 6	#13, #21 8	#13, #21 1	None
Discrete Input Processor #14 (DP)	80	80		A	--	S-IC Outboard Engine Out	Single Execution	--	Major Loop	6	8	1	None
Discrete Input Processor #15 (DP)	13	13		A	--	S-II AFT Interstage Separation	Single Execution	--	Major Loop	0	0	0	None
Discrete Input Processor #19 (DP)	17	17		A	--	S-II Engines Out	Single Execution	--	Major Loop	6	8	1	None
Discrete Input Processor #21 (DP)	112	112		A --		S-II Outboard Engine Out	Single Execution	--	Major Loop	6	8	1	None
Discrete Input Processor #22 (DP)	499	499	#22, #28 186	A	--	S-IVB Cutoff "B" Discrete	Single Execution	--	Major Loop	#22, #28 10	#22, #28 37	#22, #28 1	385

Table 4-1
(continued)

SATURN V BOOST TASK PROFILE

Task Name/ID	Execution Time (add equiv) Nominal	Execution Time (add equiv) Maximum	Memory Req. (# of instr.)	Synchr/ Asynchr	Iteration Rate	Activate Upon	Deactivate Upon	Preceding Task	Completion Deadline	Parameters Input	Parameters Output	Telem. (words per equiv)	Inter. Inhb. (add equiv.)
Discrete Input Processor #28 (DP)	507	507		A	--	S-IVB Cutoff "A" Interrupt	Single Execution	--	Major Loop	10	37	1	385
F/M Calculations (DV)	128	130	96	S	ML	TB1+6 TB3+6.7 TB4+12.0	TB3+0 TB4+0 TB5+0	AP	Major Loop	10	7	3	None
Minor Loop Error Monitor (EC)	70	70	90	S	1.25/ sec	Phase Start	Phase End	UM	800 msecs	8	10	0	None
Gravitation Acceleration (GR)	300	300	160	S	ML	Phase Start	Phase End	NE	Major Loop	7	6	0	None
S-IVB Cutoff Prediction (HS)	238	353	201	S	ML	TB5-8 If Velocity Sufficient	TB5+0	IG	Major Loop	24	7	3	28=NOM 33=MAX
Iterative Guidance Mode (IG)	3276	4115	1477	S	ML	7B3+40.6	TB5+0	NE	Major Loop	45	11	36	None
Minor Loop (ML)	199	522	1167_8	S	25/sec	Phase Start	Phase End	--	40 msecs	46	33	1	Duration of Execution
Minor Loop Support (MS)	458	458	199	S	ML	Phase Start T5+0	Start HS Phase End	CC or TT	Major Loop	14	10	10	13+30 = 43

Table 4-1
(continued)

SATURN V BOOST TASK PROFILE

Task Name/ID	Execution Time (add equiv) Nominal	Execution Time (add equiv) Maximum	Memory Req. (# of instr.)	Synchr/ Asynchr	Iteration Rate	Activate Upon	Deactivate Upon	Preceding Task	Completion Deadline	Parameters Input	Parameters Output	Telem. (words per equiv)	Inter. Inhb. (add equiv.)
Boost Navigation (NE)	689	829	287	S	ML	Phase Start	Phase End	AP	Major Loop	14	18	16	None
Digital Output Multiplexer Telemetry (OD)	848	848	75	S	ML	TB5+0	Phase End	--	Major Loop	1	1	4	282+261+ 213 = 756
Orbital Guidance (OG)	582	606	487	S	ML	TB5+0	Phase End	NE	Major Loop	27	20	2	None
Time-To-Go To Restart, Beta Test (RS)	527	737	242	S	ML	TB5+0	Phase End	NE	Major Loop	15	8	5	None
Steering Misalignment Correction (SM)	299	299	77	S	ML	TB3+60.6 TB4+15.0	TB4+0 TB5+0	IG	Major Loop	10	2	2	None
Switch Selector Processing (SS)	431	911	853	A	--	Phase Start Scheduled for execution each time an SS command is issued.	Phase End	--	100 msecs	10	7	2	Duration of Execution
M/F Smoothing (ST)	124	124	91	S	ML	TB3+6.7 TB4+12.0	TB4+0 TB5+0	DV	Major Loop	17	2	2	None
Time Base Start Routines #00 (TB)	275	314	28	A	--	Start of each time base	Single Execution per TB	--	Complete <2 msec after act. cond detected	2	6	0	Duration of Execution

Table 4-1
(continued)

SATURN V BOOST TASK PROFILE

Task Name/ID	Execution Time (add equiv) Nominal	Maximum	Memory Req. (# of instr.)	Synchr/ Asynchr	Iteration Rate	Activate Upon	Deactivate Upon	Preceding Task	Completion Deadline	Parameters Input	Output	Telem. (words per equiv)	Inter. Inhb. (add equiv.)
Time Base Start Routines #10 (TB)	393	432	#10, #15 17	S	ML	TO+17.5	Liftoff	--	Complete <2 msec after act. Cond detected	#10, #15 8	#10, #15 7	#10, #15 1	Duration of Execution
Time Base Start Routines #15 (TB)	337	376		A	--	Liftoff Interrupt	Single Execution	--	Same	8	7	1	Duration of Execution
Time Base Start Routines #25 (TB)	321	360	29	A	--	T1+134.7	Single Execution	--	Same	3	4	0	Duration of Execution
Time Base Start Routines #30 (TB)	352	391	#30, #35 44	A	--	S-IC Outboard Engine Out Inter	Single Execution	--	Same	#30, #35 3	#30, #35 2	#30, #35 2	Duration of Execution
Time Base Start Routines #35 (TB)	339	378		A	--	Ref. Table 4-2	Single Execution	--	Same	3	2	2	Duration of Execution
Time Base Start Routines #40 (TB)	388	427	#40, #45 56	A	--	Ref. Table 4-2	Single Execution	--	Same	#40, #45 4	#40, #45 5	#40, #45 2	Duration of Execution
Time Base Start Routines #45 (TB)	375	394		A	--	Ref. Table 4-2	Single Execution	--	Same	4	5	2	Duration of Execution

Table 4-1
(continued)

SATURN V BOOST TASK PROFILE

Task Name/ID	Execution Time (add equiv)		Memory Req. (# of instr.)	Synchr/ Asynchr	Iteration Rate	Activate Upon	Deactivate Upon	Preceding Task	Completion Deadline	Parameters		Telem. (words per equiv)	Inter. Inhb. (add equiv.)
	Nominal	Maximum								Input	Output		
Time Base Start Routines #50 (TB)	64	64	#50, #55, #57 156	A	--	Ref. Table 4-2	Single Execution	--	Complete <2 msec after act. Cond De- tected	#50, #55, #57 7	#50, #55, #57 13	#50, #55, #57 3	Duration of Execution
Time Base Start Routines #55 (TB)	54	54		A	--	Ref. Table 4-2	Single Execution	--	Same	7	13	3	Duration of Execution
Time Base Start Routines #57 (TB)	62	62		S	ML	T4+10.0	T5+0	--	Same	7	13	3	Duration of Execution
Time Tilt Guidance (TT)	94	140	211	S	ML	TB1+0	TB3+0	--	Same	5	2	0	None
Time Update (UM)	105	105	11	S	1.25/ sec	Phase Start	Phase End	--	800 msec	1	0	0	None
Utility Routines	--	--	534	--	--	--	--	--	--	-	-		

Table 4-1
(continued)

SATURN V BOOST EVENT SEQUENCE TIME LINE

Name	Time Base (TB)	Time in Time Base (Secs)			Start Condition	TB Start Routine
		Nom.	Min.	Max.		
Guidance Reference Liftoff	0				Interrupt 7	
Liftoff	1	17.0	16.0	150.0	Discrete 24 (after TB0+16) or vertical inertial acceleration<2 m/sec^2 for 4 comp. cycles after 17.5.	TB15
S-IC Inboard Engine Cutoff	2	134.7			TB1 + 134.7 if down range velocity <500 m/sec.	TB25
S-IC Outboard Engine Cutoff	3	29.7	18.4		Interrupt 5 or discrete 18 (neither recognized until after S-IC Outboard Engine Cutoff Enable Switch Selector issued at TB2 + 18.4).	TB30 (Interrupt) TB35 (Discrete)
S-II Cutoff	4	391.9	355.0		Interrupt 6 or discrete 19 (neither recognized until after S-II LOX Depletion Sensor Cutoff Arm Switch Selector issued at TB3 + 355.0).	TB40 (Interrupt) TB45 (Discrete)
S-IVB First Cutoff	5	500.0	10.0		Any combination of 2 of the following events after TB4 + 10.0. 1. Interrupt 4 2. Discrete 5 3. S-IVB cutoff command issued by LVDC. 4. Velocity change less than 1 m/sec over the last major loop.	TB50 (Interrupt) TB55 (Discrete)

Table 4-2

APPENDIX 1

MAP TABLE FOOTNOTES

No.	Footnote
1	The majority of the references reviewed assumed a maximum word length of 32 bits.
2	Reference number SS-4 (Appendix 2).
3	In reference number SS-4 (Appendix 2) there were 7941 data words while the minimum core total (for orbital free flight) was 21,474 words. This defined a data/total core ratio of 36.9%. This ratio was then used to calculate all mainframe data estimates associated with reference SS-4.
4	Reference number M-1 (Appendix 2).
5	Reference number AS-1 (Appendix 2).
6	Reference number AS-4 (Appendix 2).
7	Using the data/total core ratio defined in footnote (3) a total core requirement of 9500 words for worst case guidance can be broken down into 3500 data words and 6000 instructions.
8	For the worst case guidance task (Guidance C) and the estimate from source AS-4 the maximum/minimum instruction ratio was 1.69 (3554 versus good). This ratio was applied as a multiplier to the other guidance minimum instruction counts as a multiplier to estimate the maximum values.
9	Reference number SS-5 (Appendix 2).
10	Reference number SS-5 (Appendix 2). The estimates are further broken down in the reference material.
11	The percentage breakdowns available in reference SS-5 were also applied here.
12	Reference number AS-1 (Appendix 2).

APPENDIX 1

MAP TABLE FOOTNOTES
(continued)

No.	Footnote
13	Reference number SS-5 (Appendix 2) was assumed to define estimates equal to the worst case control operation.
14	The data/total core ratio of footnote (3) equal to 36.9% was used to breakdown the total estimate available.
15	Reference number SH-8 (Appendix 2) defined 5844 instructions to require 4333 words of memory and 250 data quantities to require a 125 memory words. However, for the purpose of obtaining worst case estimates this report treats each item as a full word. In this case the data reference was not used.
16	Reference number M-4 (Appendix 2). Only the data estimate was used.
17	Reference number SH-8 (Appendix 2). For worst case estimating it was assumed that each data and instruction quantity required a full memory word.
18	Reference number SS-3 (Appendix 2).
19	Reference number M-4 (Appendix 2).
20	The iteration rates for reference number SH-4 (Appendix 2) were estimated by dividing one half of the instruction count into the estimate for operations/second defined. This assumes that in general about 50% of the instructions of a functional area are executed per iteration.
21	Reference number SH-2 (Appendix 2).
22	Reference number SH-8 (Appendix 2).
23	Reference number SH-5 and SS-1 (Appendix 2).
24	Calculated by dividing the instruction counts by two. See footnote (20).

APPENDIX 1

MAP TABLE FOOTNOTES
(continued)

No.	Footnote
25	An M&S Computing Engineering Judgement.
26	The majority of references reviewed use 16 or 32 bit words.
27	Calculated by multiplying the operations/iteration defined in the task hardware parameters times the defined iteration rate.
28	Extracted from the Task Hardware parameters.
29	Transferred from the phase task tables.
30	Assumed equivalent to insertion
31	Assumed equivalent to orbital free flight (OFF).
32	Assumed equivalent to Rendezvous Targeting.
33	Assumed equivalent to Powered Ascent.
34	This is the sum of the Interplanetary extremes components.
35	The Interplanetary phase extremes were used.
36	Reference number SH-7 (Appendix 2). The guidance and navigation estimate was split.
37	Reference number SH-7 (Appendix 2).
38	Not applicable.
39	Reference number SS-5 (Appendix 2) was the only reasonable estimate documented. The estimate of reference number SH-7 obviously included subsystem functions other than just the G&CC.
40	Reference number SS-2 (Appendix 2).

APPENDIX 1

MAP TABLE FOOTNOTES
(continued)

No.	Footnote
41	Reference number M-1 (Appendix 2). It was assumed that the video data was not processed by the on-board computer.
42	Calculated by averaging percentage over the core required for instructions. (The minimum and maximum core estimates for instruction were added together.)
43	Reference number SH-2 (Appendix 2) indicates that 75.4% of the programs are common between phases,therefore, the overall estimate was calculated by multiplying the worst case estimate/column times 1.327 to obtain an overall estimate.
44	Used the phase hardware parameter Interplanetary extremes.
45	Calculated by averaging the phase sums.
46	Used the worst case estimate/column.
47	Reference number SH-7 (Appendix 2).
48	Reference number SS-2 (Appendix 2).
49	Used the data/total core ratio of 36.9 (footnote (3)) and the minimum total of 34,000 from reference number SS-3 (Appendix 2).
50	Used the data/total core ratio of 36.9 (footnote 3) and the total of 121000 from reference SH-7 (Appendix 2).
51	The totals were calculated using the following formula: Total = (Boost/OFF/OPF) + 24.6% of Interplanetary + Experiments The above formula is based on the assumption that 75.4% of the programs required for the Interplanetary phase are already included in the Boost/OFF/OPF estimate (see footnote 43).
52	Reference number SS-9 (Appendix 2).

APPENDIX 2

REVIEWED REFERENCES

I. SHUTTLE

SH-1 Data Management System - System Architecture Data Bus Computer Design Software. June 11, 1971, TRW.

SH-2 Data Management System Control Study - Technical Design Notes 1 - 21. May to September 1971, TRW.

SH-3 Software Presentation Agenda. June 1971, TRW.

SH-4 Participation in Phase B Space Shuttle Program Definition. March 2, 1971, IBM.

SH-5 On-Board Autonomy Panel Report, February 4, 1970, MSC, KSC, MSFC.

SH-6 Presentation on Low Cost Shuttle, March 1971, MSFC.

SH-7 System Configuration and Executive Requirements Specification for Reusable Shuttle and Space Station/Base. May 1971, Computer Sciences Co.

SH-8 Phase B Shuttle Review. September 1970, IBM.

II. SPACE STATION

SS-1 On-Board Autonomy Panel Report. February 4, 1970, MSC, KSC, MSFC.

SS-2 Modular Space Station Phase B Extension Second Quarterly Review. August 19, 1971, North American Rockwell.

SS-3 Pre-Phase A Study for an Analysis of a Reusable Space Tug. March 22, 1971, North American Rockwell.

SS-4 Modular Space Station Phase B Extension Third Quarterly Report. November 4, 1971, North American Rockwell.

SS-5 Reconfigurable G&C Computer Study for Space Station Use. January 31, 1971, Autonetics.

REVIEWED REFERENCES
(continued)

SS-6 Space Station Program Extension Period Second Performance Review. July 1971, MDAC.

SS-7 An Engineering Study of On-Board Checkout Techniques. March 1971, IBM.

SS-8 Final Report Multiprocessor Computer System Study. March 1970, Intermetrics.

SS-9 Modular Space Station Computer Study. October 1971, IBM.

III. APOLLO/SATURN

AS-1 STS Software Development (Study Task 5). July 1970, MIT, Charles Stark Draper Laboratory.

AS-2 Guidance, Navigation and Control CM Functional Description and Operation using Flight Program Colossus 2c (Comanche 67). August 1969, MIT Instrumentation Laboratory.

AS-3 Apollo 14 (January 31, 1971) AS-509/CM-110/LM-8 Preliminary Flight Plan. September 23, 1970, MSC.

AS-4 Saturn LVDC Statistics. February 8, 1971, MSFC.

IV. MISCELLANEOUS

M-1 Study of Spaceborne Multiprocessing. September 1968, Autonetics.

M-2 Reliability in Long-Life Missions. October 1970, Jet Propulsion Laboratory.

M-3 HEAO Computer Requirements. August 1971, M&S Computing, Inc.

APPENDIX 2

REVIEWED REFERENCES
(continued)

M-4 Data Subsystems for a 12-Year Missions. September 1970, JPL.

M-5 Final Report Voyager Spacecraft Phase B, Task D (Volume IV). October 1967, General Electric.

M-6 Reference Earth Orbital Research & Application Investigations (Blue Book). January 1971, General Dynamics.

M-7 Experiment Requirements Summary for Modular Space Station and Space Shuttle Orbital Applications and Requirements. April 1971, Martin Marietta.

SECTION 4

SYNCHRONOUS VERSUS NON-SYNCHRONOUS SCHEDULING CONTROL

The task scheduling and interrupt handling approach selected has far-reaching effects on the overall design of a computer system such as ARMMS. The two major alternatives are synchronous and non-synchronous scheduling. This section discusses these alternatives, and the conclusion is made that non-synchronous scheduling should be selected for ARMMS. The design is proceeding based on this conclusion.

SYNCHRONOUS VS. NON-SYNCHRONOUS SCHEDULING CONTROL

I. INTRODUCTION

A controversy currently exists concerning the methods of task scheduling to be used for aerospace systems. This controversy exists due to a revival of synchronous scheduling methods. Synchronous scheduling methods were revived as an apparent solution to prevent subtle but critical software errors seemingly caused by the use of so-called interrupt driven-scheduling methods (non-synchronous scheduling).

There is no question that synchronous scheduling methods are intuitively appealing. However, in the thus generated enthusiasm the shortcomings are likely to be ignored. It is the purpose of this report to highlight these disadvantages, such that the merit of synchronous scheduling control is evaluated with the proper perspective.

In the following paragraphs we will first present the aerospace task scheduling characteristics, then describe the functional design of synchronous scheduling control and finally discuss the pros and cons of the subject scheduling methods.

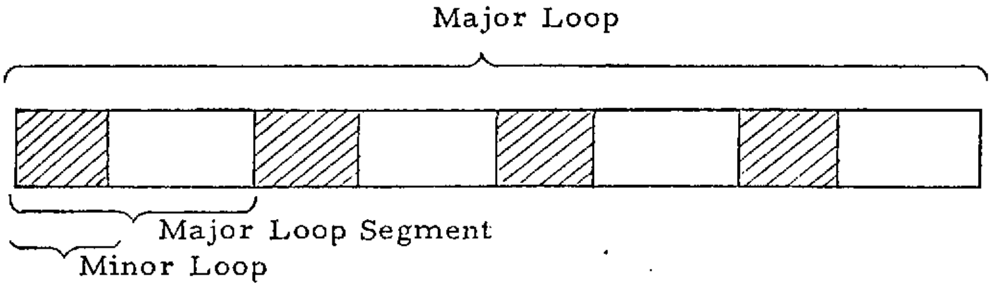
II. AEROSPACE SOFTWARE SCHEDULING CHARACTERISTICS

Aerospace software consists first of all of a set of cyclic processes, each of which may require a different repetition rate. These are generally called the <u>synchronous processes.</u> The highest repetition rate required is called the <u>minor loop frequency</u> and the time required to perform the processes to be executed at this rate is called the <u>minor</u> loop. The time available between consecutive minor loop start points is called a <u>major loop segment.</u> The shortest time span in which the majority of the synchronous processes can be executed at least once, is called the <u>major loop.</u>

For example, if there are three task with the following characteristics:

Task Name	Rep. Rate	Exec. Time
A	8 per sec.	50 msecs.
B	4 per sec.	75 msecs.
C	2 per sec.	150 msecs.

Then the minor loop frequency is 8 per sec. and the minor loop is 50 msecs. A major loop segment is 1/8 sec. or 125 msecs., and the major loop is 1/2 sec. or 500 msecs. This is further depicted below:



Note that there may be (generally non-critical) supporting processes that, in effect, have a higher repetition rate than the minor loop. An example of this in the Saturn V flight program is telemetry. A piece of data can be transmitted every 4.3 msecs.

Secondly, there is a set of processes that are forced by non-periodic events such as a specific time, an event signalled by the external environment, or other conditions that could be detected during execution of another process.

These are called asynchronous processes. Some of these processes are time-critical. That is, they have to be completed within the next two or three major loop segments. Some of these processes are non-time critical. That is, their execution may be delayed several major loop cycles if necessary. There is, of course, a range of response requirements between the defined time-critical processes and non-time critical processes. The main point is that some of the asynchronous processes are likely to be time-critical. It is obvious, that if these can be properly handled, processes with lesser response requirements can certainly be accommodated.

III. SYNCHRONOUS SCHEDULING CONTROL DESIGN

For Synchronous Scheduling Control, it is necessary to divide each process into segments such that each segment is completed within a specified time. That is, the primary segmentation criterion is execution time. This may or may not correspond to the functional modularization.

The maximum execution time is dependent on several criteria which will become more apparent further on in this discussion.

During each major loop segment all minor loop processes are first of all completed. The remainder of the major loop segment is then assigned to the various process segments such that processes are executed at the required cyclic rate and the individual process segments assigned to a major loop segment are completed before the next minor loop should start.

There is no algorithm available to easily establish maximum length of time and major loop segment assignment of the various process segments. However, the following method provides a simple start that should allow proper division and assignment of the process modules to be completed within a minimum number of iterations. It is an approach that identifies the problems likely to occur with any approach taken.

1) Compute the time interval available in each major loop segment for major loop processing (difference between the major loop segment and the minor loop execution time).

2) To limit interference in assignment of processes scheduled at different repetition rates, it is desirable to be able to assign one process segment from each repetition rate in a single major loop segment. Therefore, the available major loop segment time, determined above, should be divided by the total number of required repetition rates (not including minor loop rate). This is the maximum execution time of a process segment.

3) Schedule each major loop segment such that a single segment of each process is consecutively scheduled, starting with the process with the highest execution rate and so on, until the processing requirements are completed. Where necessary adjust process segment sizes to accomplish this.

4) Rearrange the process segments such that process segmentation can be minimized if necessary by recombining segments. Note, however, that this makes the process segment assignment less flexible and therefore less adaptable to modifications.

A simple example:

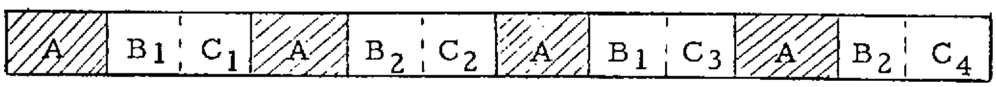
Assume the same task mix that was used before

Task Name	Rep. Rate	Exec. Time
A (minor loop)	8 per sec.	50 msecs.
B	4 per sec.	75 msecs.
C	2 per sec.	150 msecs.

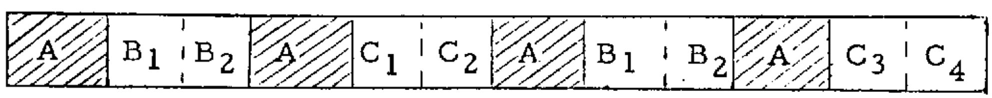
As computed before, the major loop segment is 75 msecs., and the minor loop segment is 50 msecs. Therefore, maximum module execution time = $\frac{175 - 50}{2}$ = 37.5 msecs.

Assigning the process segments to consecutive time slots results in the following major cycle assignment:

Minor Loop



Note that this is not a unique solution. The following assignment is entirely feasible, but not always immediately obvious.



Note that several segments can now be recombined.

The above described the major cycle scheduling for cyclic processes. Our next concern is how to accommodate asynchronous processes.

A possible method is to consider an asynchronous process as a cyclic process that is not always executed. The (imaginary) repetition rate is assigned by considering the response time constraints and assigning it such that the process can always be completed within these constraints. For example, if the maximum response time is two major cycle segments, the repetition rate should be considered equal to once per two major cycle segments. If the response time is any less than two major cycle segments, it should be considered a minor loop process. Note that multiple, mutually exclusive, asynchronous process may be assigned to the same subsegment of the major loop segment.

Other simpler assignments of asynchronous processes are, of course,possible. For example, depending on maximum response requirements every other, or third, or fourth major loop segment may be assigned solely to asynchronous processes. This puts a constraint, however, on the repetition rates that can be accommodated.

In summary, the steps to be performed to establish the process module scheduling are as follows:

1) Compute minor cycle, major cycle.

2) Compute maximum execution time allowed for process segments.

3) Transform asynchronous processes to apparent synchronous processes.

4) Apply scheduling assignment discussed.

5) Rearrange assignment to minimize number of process segments (optional).

IV. ADVANTAGES/DISADVANTAGES OF SYNCHRONOUS SCHEDULING CONTROL

4.1 Advantages

There are some distinct advantages of synchronous scheduling control (as opposed to non-synchronous scheduling) that should be highlighted first.

1) Program "break points" are controllable and therefore more likely to be free of errors.

The requirements to "interrupt" execution of a process in favor of execution of another process is not removed. However, these "interrupts" are break points provided in a preplanned and controlled manner by segmenting the processes. Common data read/write conflicts etc., still exist. However, they can be predicted and may be controlled by design.

2) The relationship between process segments in space and time is largely fixed and is, therefore, likely to cause less errors. With the exception of the asynchronous processes, all process segments are permanently assigned to specific major loop segments in a specific sequence. The interaction between program segments is almost fully predictable and repeatable.

   Potential errors can, therefore, be more easily found, and verification of the proper behavior of the processes under various conditions is more easily established.

3) Accuracy of repetition rates can be controlled within very narrow limits.

   Synchronous scheduling is the only known (practical) method available that allows all repetition rates to be very accurate.

   Non-synchronous scheduling schemes are usually accurate for the two highest repetition rates, but become exponentially less accurate for lower repetition rates.

   This accuracy is not always a true advantage, as accuracy for the lower repetition rates is usually not significant. However, it is certainly never a disadvantage, and may be significant for certain applications.

The above then are the advantages that the synchronous scheduling method has over non-synchronous scheduling schemes. Whether these advantages offset the associated disadvantages, is the question that remains to be answered.

4.2 Disadvantages

The major disadvantages to be discussed center around the problem of segmenting of the processes and the efficiency of the method.

4.2.1 Efficiency

During actual execution of the major cycles a large amount of spare processor time must be made available to accommodate asynchronous process and variations of execution time in synchronous processes.

The argument put forth by proponents of synchronous scheduling is that this spare capacity has to be available no matter which scheduling method is used. This is true, but only partially so. The difference is that in synchronous scheduling each major loop segment has to be able to accommodate the peak load for all modules that may execute during that segment. This is true because all processing has to be completed before the start of the next minor loop. If the major loop segment overruns its allotted time, the system becomes inoperative.

In non-synchronous scheduling, however, the process or sizing can be performed on the average loading encountered. This is true because the method has the inherent capability to recover from peak load. All that happens during peak load conditions is that the lower priority cyclic processes get slightly delayed during that period of time. That is, the periodicity is automatically slightly decreased. As soon as the peak load disappears, the system automatically returns to normal.

The difference between average load and peak load sizing is estimated at least 10% for most applications, but could range as high as 20% of the available processing power. This is significant and could become a critical factor.

4.2.2 Process Segmenting

The major advantage assigned to synchronous scheduling is that, because of the non-existence of interrupts, correct process behavior under various conditions is easier to accomplish and verify, thereby saving a considerable amount of debug time.

The disadvantages listed below tend to indicate that this advantage could very well be offset by the time required to verify major loop segment timings and additional coding time forced by the segmenting criteria of the processes.

Process Segment Timing

As discussed several times before, this becomes a highly critical issue. In non-synchronous scheduling an oversight is a minor annoyance, in synchronous scheduling it is an irrecoverable error.

In non-synchronous scheduling it is sufficient to carefully time the time critical processes under all known conditions but only check the other processes under most common conditions.

In synchronous scheduling it is necessary to carefully time each process segment under all possible conditions. This could cost a significant amount of debug time. Note also that a minor modification to a process segment could result in a major difference in execution time. This in turn could result in resegmenting and reassignment of segments and therefore result in a complete reverification of the system.

Process Segmenting

For applications of any complexity this is not a trivial matter. As will be described below, segmenting and resegmenting (and, therefore recoding) tends to be an almost continuous process from the initial design until complete verification. In addition, modifications to the verified system may again result in resegmenting (and,therefore, recoding) of several processes and reverification of the complete system.

A rational approach to the segmenting goes as follows:

1) Initially design all process segments using functional modularity.

2) Estimate process segment timings. Redefine process segmentation where sizing is obviously out of line with the objectives.

3) Assign process segments to major loop segments. Redesign segmentation where the initial design does not fulfill the objectives.

4) Code the process segments and compute times from the completed coding. This is essentially a repeat of step two. Several iterations of steps 2, 3, and 4 are normally necessary as process segments are completed.

Surely it is a normal condition that the timings obtained from the coding can significantly differ from the original estimates.

As process segments are debugged and modified, reestimating and resegmenting will take place, until sufficient confidence is obtained in the timing estimates. It is easy to see that this will increase the coding and verification effort.

Where processes must be segmented beyond their functional modularity, there is an attendant increase in the process segment interfaces. These additional interfaces also increase the design and verification efforts.

The difficulties of process segment assignment to major cycle segments is hard to appreciate for someone who has never tried it for an actual application. Some idea may be obtained by considering the primary parameters that affect the assignment:

1) Process cyclic rate.

2) Process segment.

3) Synchronization required with other processes.

4) Timings of other process segments already assigned to a major loop segment.

5) Availability of feasible break points at the proper point in the process.

Note specifically the interaction between process segment assignments. This is highly iterative process that may result in several major redesigns. Note also that the real problems do not become obvious until the application processes are designed in detail, which is accomplished much later than the detailed design of the Control Executive and the commitment to its use.

In all fairness it must be noted, however, that the disadvantages emphasized for segmenting becomes trivial if the processor is always under utilized (50-60% of available capacity) and the minor loop frequency is low, i.e., the major loop segment is long.

V. SUMMARY

From the preceding discussion we are forced to draw the following conclusions:

1) Synchronous scheduling is definitely advantageous if

   - The application processes are fairly straight forward.

   - Not more than 3 or 4 different cyclic rates have to be used.

   - It is permissible to underutilize the available processor capacity.

2) There is a total lack of evidence on which to base the critical decision that future space vehicles should use a Synchronous Control Executive. Neither will there be sufficient evidence until the implementation of the application processes is well advanced.

3) Sufficient problems have been identified for synchronous scheduling to indicate that non-synchronous scheduling is alive and well, and could still be the best approach.

We conclude that the ARRMS effort should be directed towards a non-synchronous executive based on minimizing (as opposed to deletion) of process module interrupts.

SECTION 5

PRELIMINARY ARMMS RELIABILITY FEASIBILITY STUDY

An early effort in Phase I was to make an attempt to evaluate the design approach necessary to achieve the 5-year reliability objective of 0.99. Of particular interest (because of their great impact on the design effort) were estimates of degree of subpartitioning of the modules and the number of replicates of each module class. The steps performed in making the analysis were these: 1) Four module classes (processor, I/O, main memory, task memory) were postulated and estimates of the failure rate of each were made. 2) The initial baseline architecture consisting of 6 modules of each class was used as a point of departure. 3) A model was programmed which allowed the major parameters (module failure rates, dormant failure rate, number of replicates per module class, number of subpartitions per module) to be varied. 4) Probability of survival of at least a simplex system after 5 years of operation was computed for many test cases.

Perhaps the conclusions of greatest interest are: 1) that the degree of subpartitioning required for the processor is low even for conservative values of processor failure rate, and 2) the reliability objective is feasible with number of replicates per module class on the order of six.

PRELIMINARY ARMMS RELIABILITY FEASIBILITY STUDY FOR MODE 3

Introduction and Summary

The Automatically Reconfigurable Modular Multiprocessor System (ARMMS) is required to operate in three different modes. Mode 1 requires the capability of recovery from a fault in real time, such as during launch. Mode 2 requires parallel processing for high computational ability with momentary disruption of activity. Mode 3 requires the system to be operating at least as a Simplex system at the end of a five year mission. The reliability requirement for Mode 3 is .99.

This report shows by means of a reliability math model that the requirement is feasible. It also shows various ways it can be achieved.

The Model

The system is visualized as relying heavily on redundancy, having replicate or duplicate modules of the major units (random access memory, task memory, processor, and input/output) with probable subpartitioning of the modules, appropriate switches (again, redundant if necessary), and a master controller and reconfigurer called the BOSS. The BOSS is responsible for detecting and isolating failures and performing the necessary switching to redundant operational modules or subpartitions.

For this study, the reliabilities of the BOSS and the switching hardware were assumed to be 1.0. Consequently, the present results establish upper limits on the ARMMS Mode 3 reliability. This assumption was made for two reasons. One, the design of the BOSS (with respect to redundancy) will be based on the results of studies such as this one. Two, the consideration of all the combinations of redundant modules and redundant switches complicate the model unduly for a preliminary study of this nature.

A reliability math model for the system as described herein has been derived in [1]. If the reliability of a single module for a time T is

$$R(T) = e^{-\lambda T},$$

where λ = the failure rate of the module,

then the reliability of a unit composed of the n operating modules,

$$R(T) = e^{-\lambda T}\left[1 + \sum_{j=1}^{n-1} (1-e^{-\lambda T})^j\right].^{*}$$

If in addition, the system is required to operate only a fraction of the time f, the failure rate of a module when turned off is a fraction c of the failure rate λ (when on), and each replicate has p partitions, then the reliability of the i^{th} unit is

$$R_i(T) = \left\{ e^{\frac{\lambda_i' T}{P_i}} \left[1 + \sum_{j=1}^{n-1} \frac{\left(1-e^{\frac{-c\lambda_i T}{P_i}}\right)^j}{j!} \prod_{k=0}^{j-1} \left(K + \frac{\lambda_i'}{c\lambda_i}\right)\right]\right\}^{P_i}$$

where λ_i' = the average failure rate (of the i^{th} unit) which is on a fraction f of the time

$$= f\lambda_i + (1-f)c\lambda_i$$

The reliability of the system is then

$$\prod_{i=1}^{4} R_i(T).$$

This reliability function has been programmed for use on the 265 time-sharing computer, so that one can vary the parameter values on-line. Thus, it is easy to determine the effect of each parameter on the overall reliability of the system.

---

*This is equivalent to the more familiar expression $R_{SYS} = 1-(1-R)^n$, in which one unit out of n must operate for system success.

Results

The programmed math model was used to (1) determine various feasible designs which would meet the reliability requirement, and (2) study the effect of the various parameters (failure rates, number of replicates, etc.). First, a baseline design and baseline set of parameter values were defined; then the parameters were varied about these values.

The baseline design was defined as consisting of six replicates of each unit and no subpartitioning ($p = 1$ for all units). The remaining parameters, which will be difficult to control, or not accurately known, or both, were given the following values:

$c = .1$

$f = 1.0$

$\lambda_1(\text{RAM}) = 60/10^6$ hr.

$\lambda_2(\text{JM}) = 44/10^6$ hr.

$\lambda_3(\text{CPE}) = 41/10^6$ hr.

$\lambda_4(\text{IOC}) = 5.5/10^6$ hr.

In addition, the mission time T was held fixed at 5 years (43,800 hr.) throughout the study.

The reliability of the baseline design with the above parameter values was only .62. This suggested the need for subpartitioning in at least some of the unit modules. Several more cases were investigated in order to find what combination of design and use parameters would yield the required reliability. Three cases which do meet the requirement are shown in Table I. Thus, the requirement appears to be achievable with a reasonable design under rather conservative assumptions.

The effect of the various parameters was studied by varying one parameter at a time about the baseline value. The parameter ranges used were as follows:

1. The number of replicates (n) was varied from 4 to 10.
2. The number of subpartitions per module (p) was varied from 1 to 4.
3. The off (dormant) failure rate to on failure rate ratio was varied from .001 to 1.0.
4. The fraction of time the system is on (f) was varied from .2 to 1.0.
5. The predicted unit failure rates were varied from 1/2 to twice the baseline value (λ).

The results of these parametric studies are shown graphically in Figures 1 through 5. Figures 1 and 2 concerning failure rate and failure rate ratio are of special concern, because of the difficulty in estimating these values accurately. The baseline values are the best estimates at present, and they will be improved upon as the program continues.

TABLE I: CASES WHICH SATISFY THE REQUIREMENT OF .99

CASE	FRACTION OF TIME ON	OFF-ON FAILURE RATE RATIO	UNIT								SYSTEM RELIABILITY
			RAM		TM		CPE		I/O		
			n	p	n	p	n	p	n	p	
1	1.0	.1	6	4	6	2	6	2	4	1	.9941
2	1.0	.3	7	4	7	2	7	2	4	1	.9922
3	.5	.3	6	4	6	2	6	2	4	1	.9931

RAM = Random Access Memory

TM = Task Memory

CPE = Central Processing Equipment

I/O = Input/Output

n = Number of Replicate Modules

p = Number of Subpartitions Per Module

FIGURE 1

SYSTEM RELIABILITY AS A FUNCTION OF FAILURE RATE

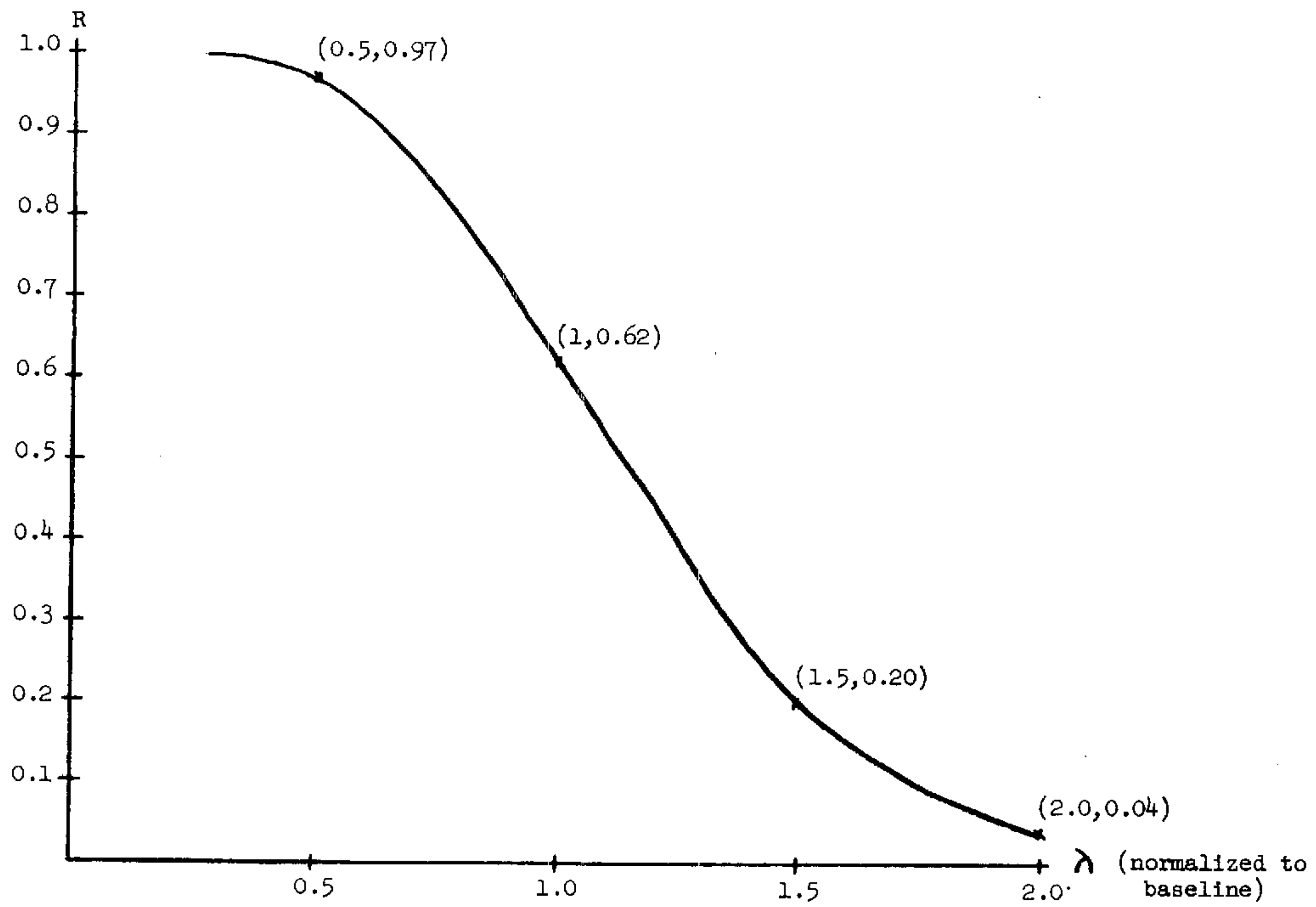


FIGURE 2

SYSTEM RELIABILITY AS A FUNCTION OF FAILURE RATE RATIO

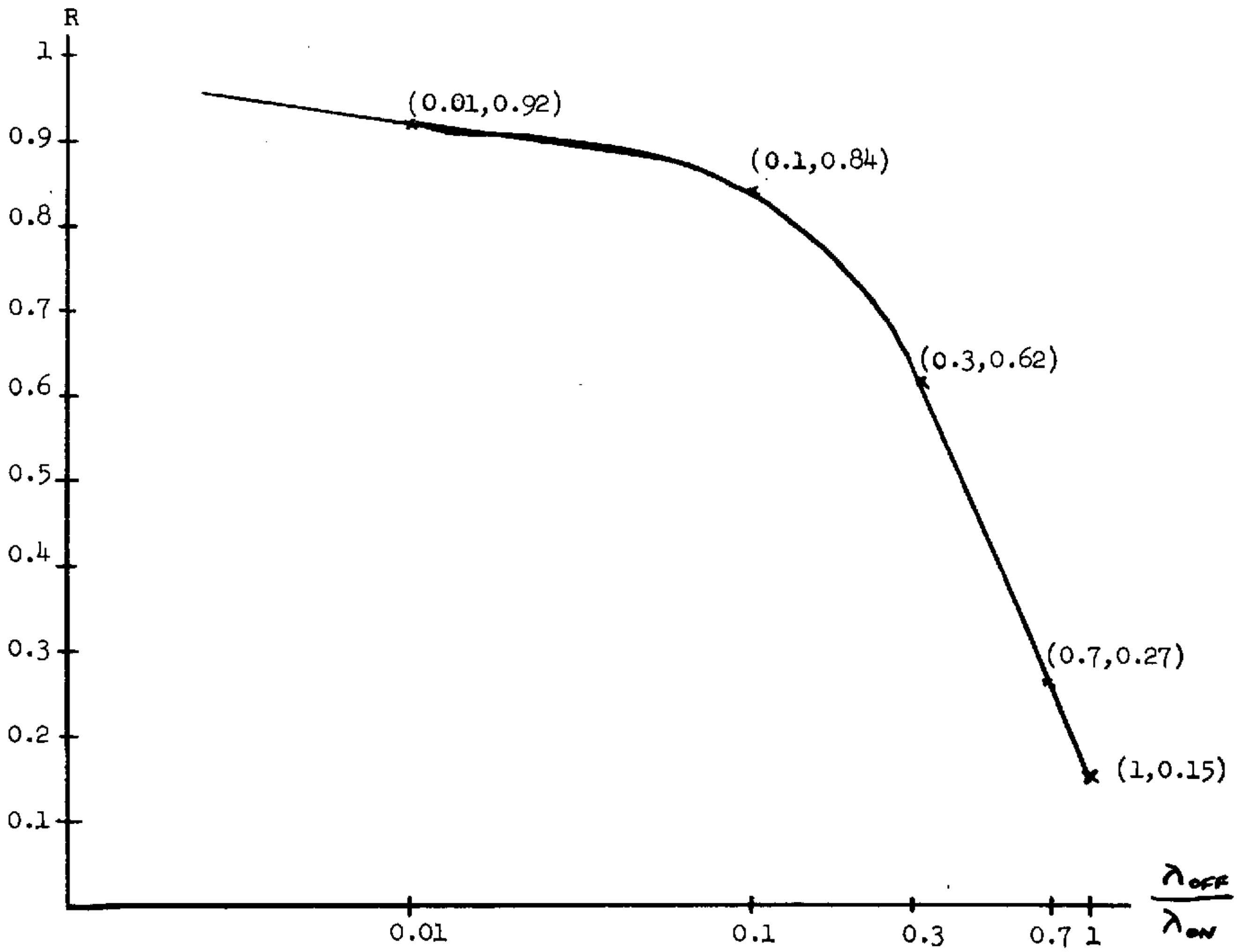


FIGURE 3

SYSTEM RELIABILITY AS A FUNCTION OF NUMBER OF REPLICATES PER UNIT

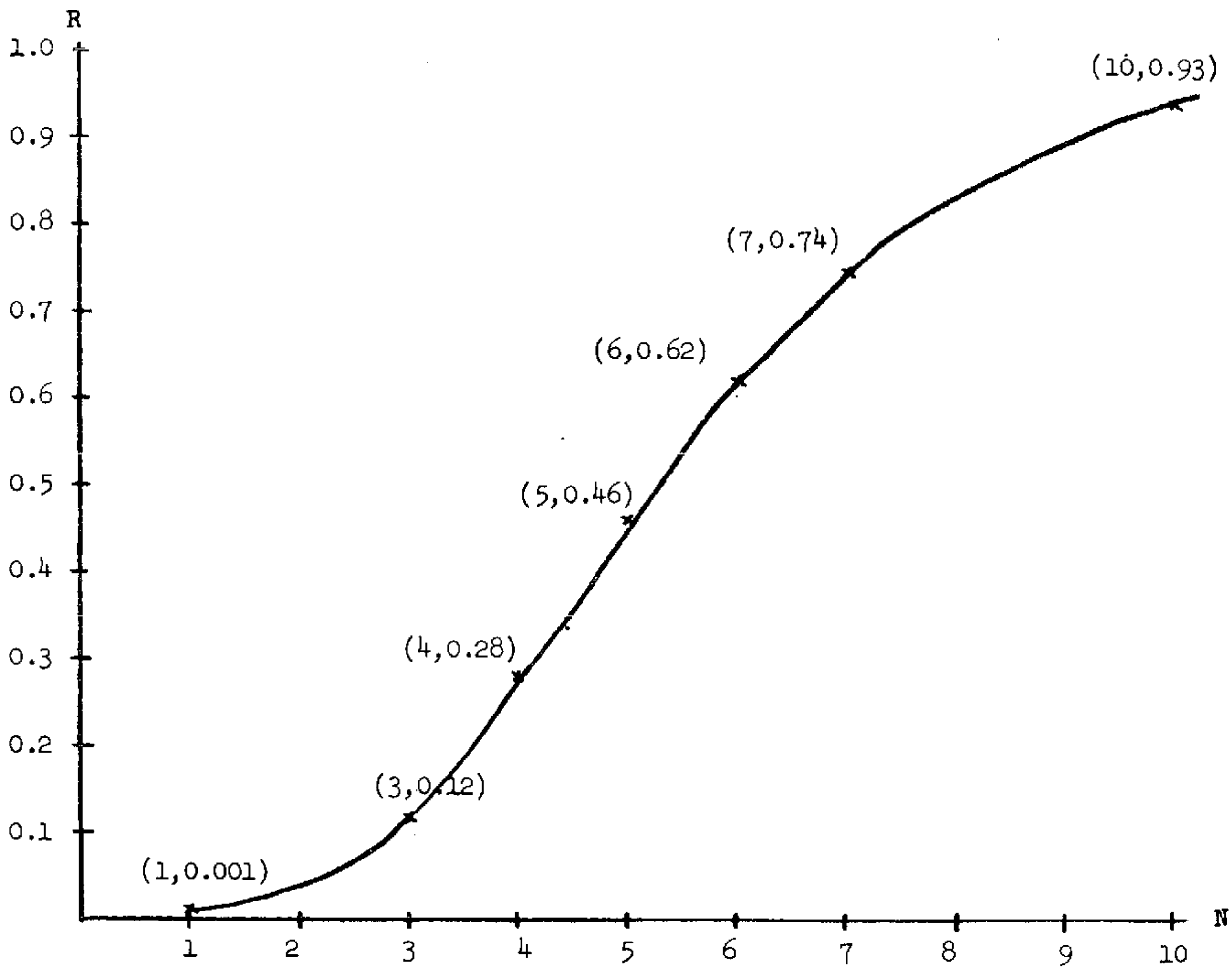


FIGURE 4

SYSTEM RELIABILITY AS A FUNCTION OF SUBPARTITIONS PER UNIT

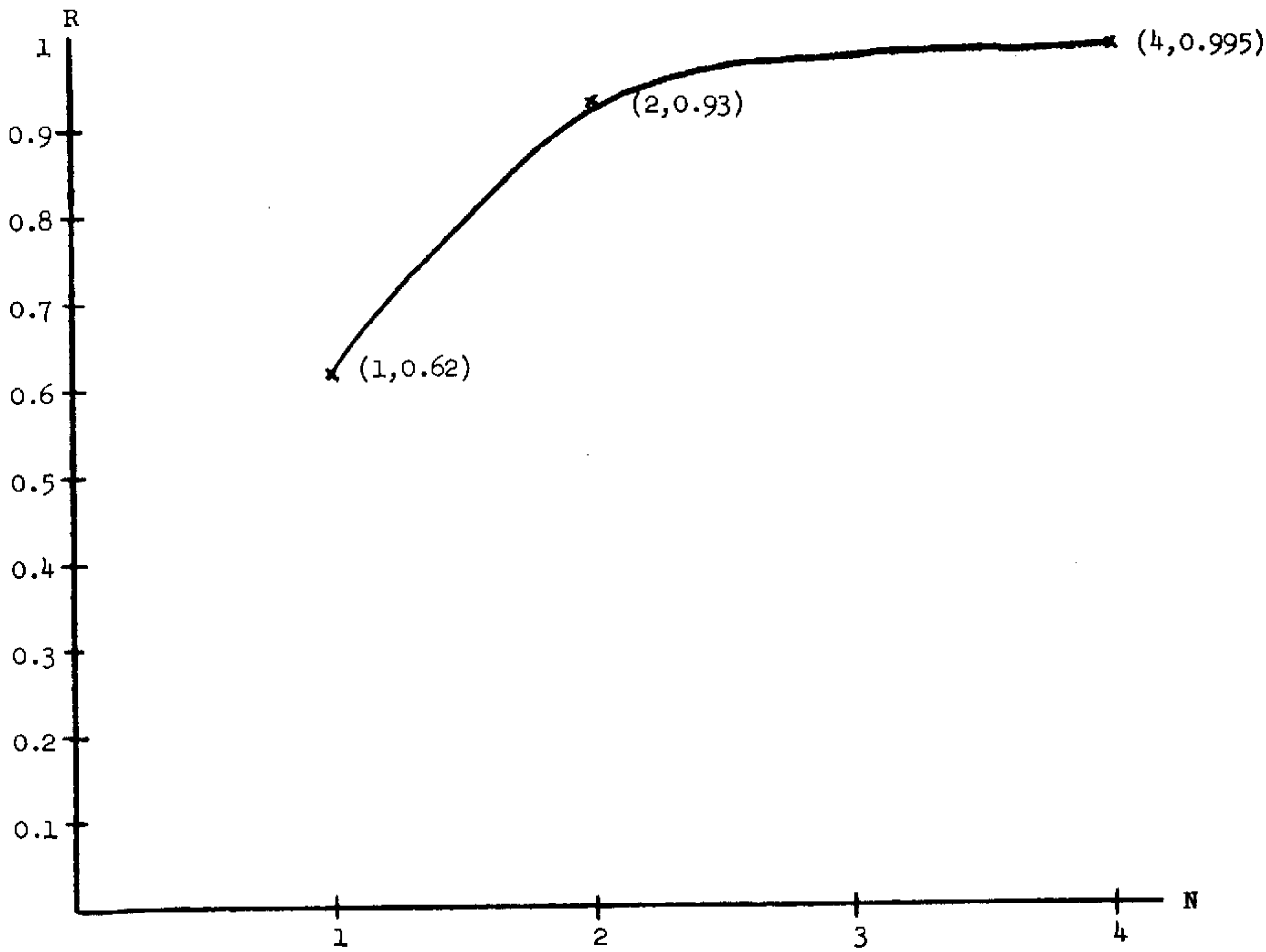
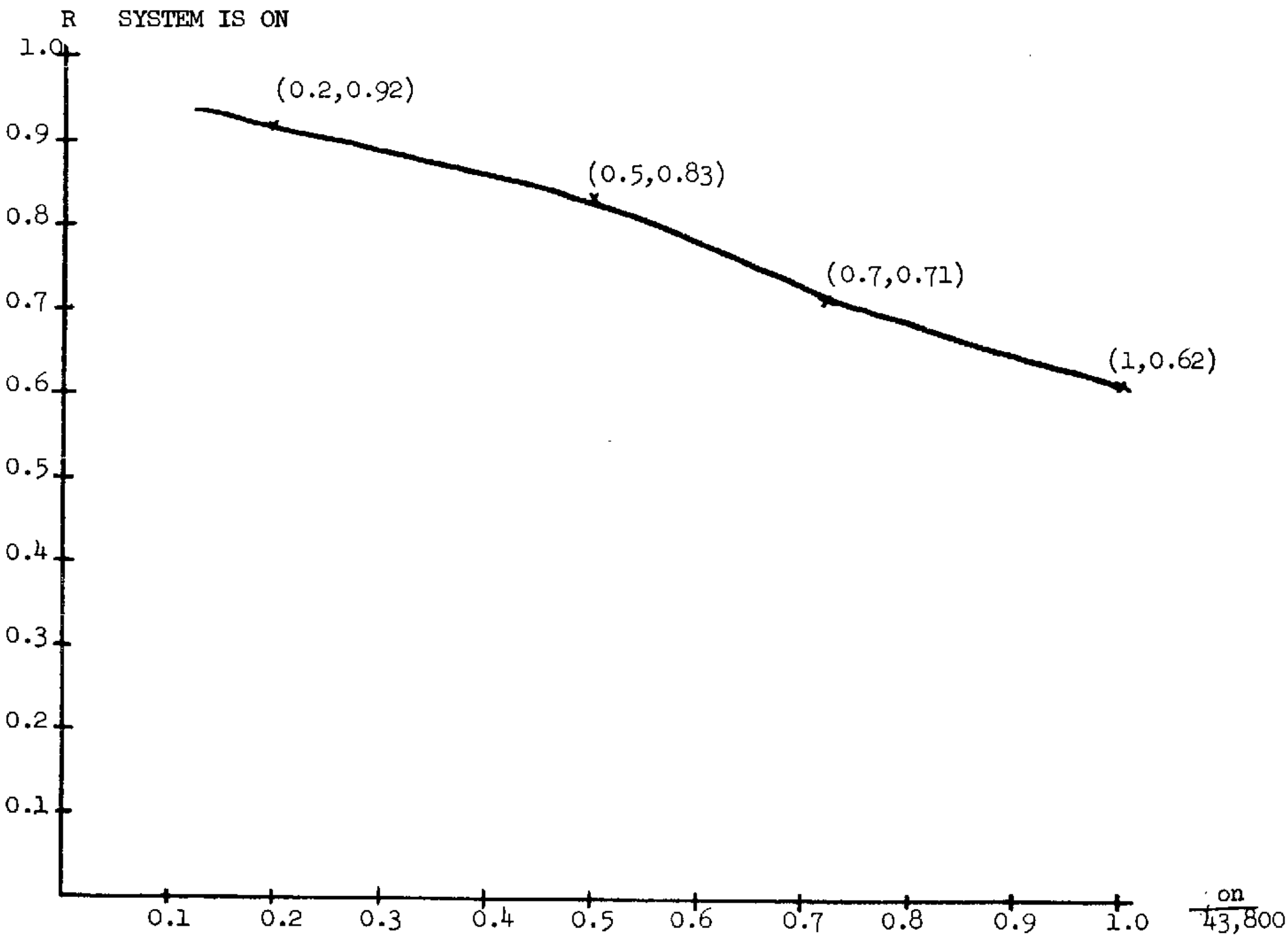


FIGURE 5

SYSTEM RELIABILITY AS A FUNCTION OF FRACTION OF TIME



REFERENCE

1. Bouricius, W. G. et al., "On-line Reliability Calculations to Achieve a Balanced Design of an Automatically Repaired Computer," NAECON Proceedings, 1967.

SECTION 6

ARMMS RELIABILITY DATA BASE

Following the initial feasibility model described in the previous section, a more thorough effort was concentrated on determining failure rates for use in subsequent modeling and design efforts. The culmination is the Failure Rate Data Base presented in Table I.

Two significant results of this report are exerting a major impact on the design. First, the microcircuit failure rates are significantly lower than those used in the initial feasibility study. Thus, processor and I/O modules require less subpartitioning to achieve a given reliability objective. Second, the ratio of dormant to active failure rate for microcircuits which appears most credible is on the order of 0.8 rather than the much lower values (typically 0.1) which have often been used. With this conclusion, the system reliability improvement provided by dormant standby units is significantly reduced.

ARMMS RELIABILITY DATA BASE

I. Introduction

A major consideration in the Automatically Reconfigurable Modular Multiprocessor System (ARMMS) program is ultra-high reliability. This will be achieved by the use of massive redundancy and by high-reliability components. This report establishes tentative component reliability data and models which will be used in subsequent reliability analyses and predictions.

The reliability parameters which this report covers are failure rates which reflect the following conditions:

1. High quality electronic piece parts for a space environment at low levels of electrical and thermal stress.
2. The above, except for power-off. This will be used for periods of dormancy and for units in a standby condition.
3. Projected 1973 technology. This is the year in which production is scheduled to begin. Most important are the projected large scale integration (LSI) microcircuit value rates.
4. The possibility of non-constant failure rates. Semiconductors and even systems often exhibit decreasing failure rates.

Each of these topics is discussed in the following material. A failure rate table is also presented.

II. Conclusions

The conclusions of the report are summarized below.

1. The failure rates to be used for ARMMS reliability analyses are handbook predictions modified by Hughes experience with space programs. These appear in Table I.
2. Failure rates for dormant units are 80% of the corresponding active failure rates.
3. The effect of power cycling on ARMMS units will be negligible.
4. LSI microcircuit failure rates are taken as 30% of the present failure rates to account for the present and anticipated rapid improvement in LSI reliability.
5. The phenomenon of decreasing failure rates in space systems should be investigated, and if substantiated, should be accounted for in subsequent reliability analyses.

III. Basic Failure Rates

Originally, the basic intent of this report was to accumulate and analyze part failure data from many space programs, and use the resulting best estimates for the ARMMS data base. This proved to be infeasible for the following reason: although there exists millions of system hours and literally billions of part hours from space programs, the data is still too scanty (as well as too uncertain) to accurately assess the failure rate of an individual part type, such as a power wirewound resistor. For instance, on the Orbiting Geophysical Observatories (OGO's), zero failures were reported on 60,000 resistors with 934,000,000 hours of operating time. The overall 50% confidence limit in this case is .0007 f./10^6 hr.

A value derived in this way has several shortcomings. One, the true resistor failure rate could easily be much higher, percentage-wise (such as .0025 $f/10^6$ hr.), or much lower (such as .0001 $f/10^6$ hr.). The amount of data, although large, is not large enough to accurately estimate such small failure rates. Two, failure rate estimates of individual resistor types, which are really what is needed, would in some cases be very high, due to the even smaller amount of data on individuals. Failure rate estimates in the zero-failure case depend heavily on the amount of time accumulated. Three, isolation of failures in an orbiting spacecraft to a piece part is often very difficult, so that many part failures do not get charged against a part, and the data is therefore questionable. Four, much of the data is several years old, and it is difficult, if not impossible, to extrapolate the figures to say, 1973.

Therefore, a different technique was used, whereby part failure rates were estimated using a combination of observed and predicted values. This was based on the two ideas: 1) that a handbook predicted value (which is partly theoretical) is better than an observed value if the data for the observed value is scanty; and 2) an estimate based on a prediction and an observation is better than one based only on observation. Furthermore, Hughes Aircraft Company has logged many hours in space with its various satellite programs, yielding useful data for reliability predictions of space systems. The expected number of failures was predicted for all satellites to date, and actual failures were then monitored, although not always classified. Thus, it was possible to compare predicted and observed failure rates at the system level, and determine the proper modifying factor for predicted failure rates. In this case, the predictions were based on the Hughes Designer's Reliability Handbook R-67-3[1], and the observed number of failures so far has been 47% of the predicted number of failures. Assuming this same relation holds for the ARMMS, a failure rate data base has been derived and appears in Table I. These failure rates reflect the following general conditions:

1. Quality: A level. This is the highest level available, requiring extensive process control, screening, and testing.
2. Electrical stress: 20%.
3. Temperature: 25°C.
4. Environment: Space flight with nearly ideal conditions and no maintenance.

In addition, various other assumptions are made for each individual component having to do with type, rating, etc.

Failure rates for plated wire memory units were taken from data published by North American Rockwell, Autonetics Division[2], and Motorola, Government Electronic Division[3]. The two sets of data agree closely.

V. Dormant Failure Rates

Surprisingly, little is known about the failure rates of parts and systems when unenergized, or dormant. Martin Marietta's study [4] on dormant failure rates is of little use because it gives no comparison between dormant and operating failure rates. The dormant failure rates given in [4] are typically worse than the values appearing in Table I, which are for power-on. Planning Research Corporation[5] has reviewed several other reports on dormant failure rates in addition to[4] and concluded that a theoretical dormant-to-operating ratio as given in [5] is best. The quoted ratios in [5] come from an Aerospace Corporation report[6]. The ratios go from 0.1 for Mil-Std parts to 0.8 for Hi-Rel parts with "rigorous specifications, stringent manufacturing controls, and extensive screening."

Using this approach and assuming that ARMMS parts will be Hi-Rel as defined above, the best available dormant-to-operating failure rate ratio is 0.8. Thus, rather than include an additional column in Table I, it suffices to say that the dormant failure rate for each part type will be 80% of the operating failure rate.

V. The Effect of Power Cycling

In addition to being affected by dormancy, electronic equipment reliability is also affected by power cycling. This subject is dealt with in detail in [5] and [8]. The general conclusion is that high rates of cycling are detrimental, but that very low rates such as 1 cy./1000 hr. have a negligible effect. However, such conclusions are not firm, because of the difficulty of measuring such a thing. For example, it is usually difficult to distinguish between a failure occurring during dormancy and one occurring at turn-on because dormant equipment is not monitored.

In view of the above, low to moderate cycling rates will be assumed to cause no failures.

VI. Projected 1973 Failure Rates

Since fabrication of the ARMMS computer is not scheduled to begin until 1973, the failure rates in this report should be somewhat lower than ones based on the recent past. Almost no good information exists on this subject, however. The approach used here is to multiply microcircuit failure rates by an additional factor of .3, and leave all others unchanged. The microcircuit failure rates in Table I include this factor.

This subject is discussed by Autonetics[7] which in early 1970 anticipated a reduction in MOS-device failure rates by a factor of 500 from 1967 to 1972, and little improvement after that. If this is so, 30% (an improvement of 3-1/3) seems reasonable (i.e., conservative) for extrapolating failure rates in 1970 (the time of the writing of the Hughes R-67-3) to 1973.

VII. Future Study Effort: Decreasing Failure Rates

So far, all the results of this report lead to fairly straightforward analytical results. Each conclusion is compatible with the ARMMS reliability models developed in [9] by J. L. Bricker. However, one aspect of long term reliability has not been discussed, and that is the possibility of decreasing failure rates (also called hazard rates). There is evidence in the reliability literature that failure rates of semiconductors and of systems continue to decrease for many thousands of hours past what we usually refer to as the "infant mortality" period (For instance, see [10],[11] and [12].) Additional work is planned in order to further substantiate this. If this is so, then it is worthwhile to derive decreasing failure rate reliability models corresponding to those in [9], which are based on constant failure rates.

Although this has not yet been done, preliminary work in this area shows that a great deal of additional complexity would be introduced. The crux of the problem lies in the fact that a history of turn-ons and turn-offs must be kept on every unit in the system. With a constant failure rate model one needs to know only the total previous time in each of the states (on and off). In a decreasing failure rate model, on the other hand, it makes a difference whether a unit is on for one hour early in its life or one hour late in its life, since its failure rate is higher at the earlier time. This problem is not too difficult in the case where all cycling is done in accordance with prescribed changes from one mission phase to another. The difficulty lies in accounting for turn-ons necessitated by units which fail at random times and must be replaced by standbys. In this case, an analysis has to account for all possible cycling histories, and the problem could quickly get out of hand. The problem may be intractable by analysis and even by simulation.

If it is felt that the aspect of decreasing failure rates is indeed worth including in the models, there is another alternative. If the difficulty lies in accounting for decreasing failure rates and on-off cycling simultaneously, then one possible approach is to omit cycling and include only the decreasing failure rate aspect. To omit cycling would be to assume that a unit has the same time-to-failure distribution regardless of whether it is on or off. For the constant failure rate case, this is equivalent to the case $\mu=\lambda$ in [9]. Although this assumption is probably not true, it may introduce less of an error than the assumption that the failure rates remain constant over long periods. Also, according to the figures quoted in Section IV of this report, the case $\mu=\lambda$ might not be too far from the truth.

Such a model would yield a second answer or approximation to the problem of ARMMS mission reliability. The computed reliability would suffer from our assuming the same time-to-failure distribution for both cases, but would be enhanced by the decreasing failure rates.

Also, the models would be very simple. For example, N-tuple modular redundancy with S spares, which is treated in [9] (p.4), has the reliability expression

$$R(N,S)[T] = \sum_{i=0}^{S+n} \binom{N+S}{i} [R(T)]^{N+S-i} [1-R(T)]^{i},$$

where $\quad n = \frac{N-1}{2}$ (N is odd), and

$R(T)$ = rel. function for one unit,

using the same notation as in [9]. For the degraded case (pp. 7-11 in [9]), where failures can occur even after all the spares are used up until only $D(< n+1)$ units are operating (but in a degraded mode), we would have

$$R(N,S,D)[t] = \sum_{j=D}^{N+S} \binom{N+S}{j} [R(t)]^{j} [1-R(t)]^{N+S-j}$$

That is, ARMMS reliability is expressible in terms of a simple binomial random variable.

The most popular decreasing failure rate model for reliability is the Weibull distribution, with reliability function

$$R(t) = e^{-\alpha t^{\beta}};\ t \geq 0,\ \alpha > 0,\ 1 > \beta > 0,$$

and hazard rate

$$h(t) = \alpha\beta t^{\beta-1}.$$

(β is restricted to values less than one in the decreasing failure rate case).

This would probably suffice for our analyses. Other models are available, however, and in particular, the Pareto distribution would probably be suitable also. This was presented in a paper by this writer at the 1971 Annual Symposium on Reliability [10]. The reliability function is of the form

$$R(t) = \left(\frac{a}{a+t}\right)^b, \; t \geq 0, \; a,b > 0,$$

and the hazard rate is

$$h(t) = \frac{b}{a+t} .$$

If this area is pursued, Hughes would

1. further substantiate the phenomenon of decreasing part and system failure rates, as mentioned earlier;
2. choose the most suitable time-to-failure distribution;
3. determine suitable estimates of the parameter values (e.g., a and b in the Pareto distribution) for the various parts and units in the ARMMS.

TABLE I: Failure Rate Data Base

PART	FAILURE RATE IN F./10^6 HR. FOR POWER ON
MICROCIRCUITS (Digital)	
100 Gate Equiv.	.012
200 Gate Equiv.	.016
300 Gate Equiv.	.020
400 Gate Equiv.	.023
PLATED WIRE MEMORY	
12K Bit	.56/K Bits
100K Bit	.14/K Bits
CAPACITORS	
Air Tr. or Gl.Q. Var.	.0006
Ceramic G.P.	.00004
Ceramic T.C.	.0007
Glass/Glass Porc.	.00008
Mica Button	.0043
Mica-Paper/Paper	.00002
Paper or Plas. Film Metalized Paper/Poly	.00004
Solid Tantalum	.00004
Non-Solid Tantalum	.0011
DIODES	
FET Current Reg.	.013
Mixer	.21
Step Recovery	.035
Tunnel	.035
Varactor	.035
Pwr. Rect.	.0004
For. Rect., Cont.	.0003
Rect. Bridge	.0011
Signal (Fast Recov.)	.0004
Signal, GP	.0003
Switching	.0003
Photo Transistor (switch)	.0007
Zener Volt. Reg.	.0008
Zener T.C. Ref.	.0009

TABLE I: Failure Rate Data Base (Cont'd)

PART	FAILURE RATE IN F./10^6 HR. FOR POWER ON
RESISTORS	
Carbon Comp.(RC42)	.0009
Carbon Comp.(RCR07)	.000005
Carbon Comp.(RCR 20,32,05)	.000013
Fixed Film	.00002
Thick Film	.0003
Fixed Shunt	.0021
Power WW	.0021
Power WW(Ch. Mount)	.0021
Prec. WW/Th. Plat.	.0009
TRANSISTORS	
NPN	
Low Noise (Audio)	.0015
Power G.P. Ampl.	.0019
RF (Power)	.0041
Switching	.0007
DC Linear	.0019
PNP	
Low Noise (Audio)	.0020
Power	.0025
Switching	.0008
DC Linear	.0025
FET	
Low Pwr. Sw.	.0061
Low Pwr. Ampl.	.0084
Low Noise (Audio Amp.)	.0107
SWITCHES	1.5

REFERENCES

1. Hughes Aircraft Co., Designers' Reliability Handbook, R-67-3, Feb. 1971.

2. Autonetics Div. of North American Rockwell, Reconfigurable G&C Computer Study for Space Station Use, Final Report, Vol. VII, 31 Jan. 1971.

3. Government Electronics Div., Motorola, Advertising Pamphlet: Low Power-High Reliability Plated Wire Memory System.

4. Martin Marietta Corp., Dormant Operating and Storage Effects on Electronic Equipment and Part Reliability, RADC-TR-67-307, July 1967.

5. Planning Research Corp., Reliability Data from In-Flight Spacecraft; 1958-1970, PRC R-1453, 30 Nov. 1971.

6. The Aerospace Corp., Failure Rates of Non-Homogeneous Parts Populations, Report No. TOR-0172(2133)-1, 15 Sept. 1971.

7. Autonetics Div. of North American Rockwell, Advanced Computer Dormant Reliability Study, 14 Oct. 1967 (N69-33064).

8. Planning Research Corp., The Effects of Ground Storage Space Dormancy, Standby Operation, and On/Off Cycling on Satellite Electronics, PRC R-1435, 28 May 1970.

9. J. L. Bricker, A Unified Method for Analyzing Mission Phase Reliability for Standby and Multiple Modular Redundant Computing Systems Which Allows for Degraded Performance, ARMMS Engineering Notebook, Hughes Aircraft Co., GSG, 6 Jan. 1972.

10. TRW Systems Group, Reliability Prediction and Demonstration for Missile and Satellite Electronics, RADC-TR-68-281, Nov. 1968.

11. B. H. Caldwell, "An Engineering Approach to Long-Life Complex Space Systems," Proceedings of the 1968 Rel. Main. Conference, pp. 2-10.

12. Autonetics Div. of North American Rockwell, Final Summary of Matrix Data, Minuteman High Reliability Component Parts, 30 Jan. 1964.

13. J. H. Engleman, "A Bayesian Time-To-Failure Distribution," Proceedings of the 1971 Annual Symposium on Reliability, pp. 350-355.

SECTION 7

RESULTS OF THE INITIAL PARTITIONING STUDY

OF THE SUMC PROCESSOR

At the inception of the program, Hughes was provided with detailed design documentation of the MSFC in-house processor. The purpose was to explore various approaches to subpartitioning the processor as needed to achieve the reliability objectives. The results of the partitioning study are summarized in the first section of the report. At present, the design is proceeding under the guideline that no subpartitioning of the processor will be necessary.

Results of the Initial Partitioning Study of the SUMC Processor

SUMMARY

An initial partitioning study of the SUMC processor has been made in order to identify the approaches which might ultimately be used in partitioning to achieve high system reliability. Preliminary reliability analysis concluded that the number of switchable partitions of the processor need not be greater than 2 to 4, and this was considered as a design guideline. A second major objective is to use as much of the existing SUMC processor design as possible.

Before attempting to partition the SUMC Processor, several existing processor partitions were examined. This was done to identify approaches which have been considered viable by practitioners in this field and to determine if any of these approaches are applicable to SUMC. The partitioning levels being sought to attain ultrareliability can also be used as a basis for comparison with SUMC.

The three processors examined were the IBM-MARCS, NASA-MCB, and JPL-STAR, all of which took the functional approach to partitioning. Therefore, vertical partitioning and internal redundancy were examined to determine what the drawbacks were which eliminated them from consideration as partitioning approaches for the processors reviewed. The investigation was also intended to uncover any advantages these approaches may afford which would be relevant to SUMC. Both of these alternatives, it was found, have the disadvantage of increasing the total processor gate count due to the internal functional replication necessary.

The first attempt at partitioning the SUMC processor therefore was the standard functional approach of separating the control function from the arithmetic unit. A second partition was made because there was an apparent size, and therefore reliability, inbalance. Total switchable interconnections mushroomed to 431 and 526 respectively from 109 for the unpartitioned processor and compared very unfavorably in this regard with the processors examined.

Several recommendations have been made to reduce the number of switchable interconnections. These recommendations can be implemented individually or in combination, and would reduce the pin count to as low as 29, with penalties incurred in speed and gate count. The recommendations are:

1) No more than two sub-partitions should be attempted;
2) Use byte interface for data crossing switchable interfaces, rather than full-word.
3) Use internally redundant control and arithmetic units to to reduce switching required;
4) Redesign to reduce control signals and
5) Eliminate the I/O interface from the CPE.

A final configuration cannot be determined until several suggestions received from MSFC have been examined, a clearer definition of the system architecture is developed and a reliability analysis of the alternatives is made. This effort will be undertaken in Phase II of the program.

C4

SURVEY OF EXISTING PROCESSOR PARTITIONS

The three processors which were examined must first be put into perspective as to the purpose of each design, the state of the design, and the documentation available in order to establish a true basis for comparison with a partitioned SUMC.

The IBM-MARCS computer is a paper design which was used as a research vehicle to examine design concepts for fault detection. The documentation consists of something less than a functional specification. The machine was specified well enough to establish reliability models based on features of the S/360 and 4 π series.

The NASA Modular Computer (MCB) has been breadboarded to test its design concepts and multi-mode capability. Because of its multi-mode capability it is closest in configuration to the ARMMS and the available documentation is more complete than for either of the other computers examined.

The JPL-Self Test and Repair (STAR) computer has been in development for several years and has an operational prototype which has undergone modifications geared to improving its reliability and producibility. It also has been designed for a specific mission. The STAR would then be expected to represent a more realistic approach than research projects. This attitude must be tempered however by its low computing capacity and the state of the technology it was designed to. The documentation available consists of published papers which are rich in architectural discussion but lacking in implementation details.

ARCHITECTURAL FEATURES OF THE IBM-MARCS

The MARCS computer is functionally partitioned into nine basic units which are interconnected by a system of four common busses as shown in Figure 1. Stand-by spares are connected to the same busses and are switched in to replace failed elements. The processing capability is limited to that of a uniprocessor and spares are used strictly for reliability enhancement. The units which comprise an equivalent central processing element (CPE) in the ARMMS configuration are: the Arithmetic Logic Unit (ALU), Program Control Unit (PCU), and Control Store (CS). The ALU and PCU communicate with the bus system which is under control of the Bus Control Unit (BCU). The BCU is used for synchronization and race control and as such was considered as a BOSS function. The CS is controlled by the PCU and communicates directly with the processor partitions as well as the BCU and the IOP. Control paths were not specified in the documentation and therefore pin counts used for comparison to SUMC will be lower bounds determined by bus connections.

ARCHITECTURAL FEATURES OF THE NASA-MCB

The NASA-MCB computer is functionally partitioned into six units which are interconnected by configuration control switches as shown in Figure 2. Spares are added by adding columns to the matrix as shown in Figure 3. A working computer can be configured as long as one copy from each row can be interconnected. The switches have the capacity for simultaneous interconnections in order that the spares, required for long term reliability, can be utilized in a multiprocessing mode to satisfy varying computational requirements. Because of this multi-mode capability the MCB most closely resembles the ARMMS configuration. It is, nevertheless, still quite different from ARMMS.

The breadboard of the modular computer consists of two parallel processors and a CAU; it therefore did not address itself to a voting configuration although an expanded version could be adapted for voting. It also did not consider replication of the CAU which grows linearly (at least in pins) with the number of processors controlled.

The Control Unit (CU) and Arithmetic Unit (AU) of the MCB would comprise the equivalent of a CPE in the ARMMS configuration. The partitioning of these two units is very inbalanced. The CU performs all control operations, logic operations and branching operations. The AU performs only the arithmetic operations, including floating point operations, under the control of the CU. The Configuration Assignment Unit (CAU) controls the inter-partition switches and establishes cross communication between parallel processors. The CAU is not the executive, however, but assigns one of the active processors (CU's) as executive.

ARCHITECTURAL FEATURES OF THE JPL-STAR

The JPL-STAR is functionally partitioned into eight units which are interconnected by a system of common busses as shown in Figure 4. Stand-by spares are connected to the same busses and are switched in to replace failed elements. Its overall architecture is therefore very similar to the to the MARCS computer and like the MARCS it is limited in capacity to a uniprocessor. It differs from the MARCS in the width of the busses, level of partitioning and interconnection of control memory.

The CPE consists of the Control Processor (COP), Logic Processor (LOP), Main Arithmetic Processor (MAP), and the Read Only Memory (ROM). The LOP's shown in the block diagram are actually dual units with disagreement detection since logic operations do not preserve arithmetic error codes which are used throughout the rest of the machine. Partitioning schemes which split logic and arithmetic operations such as the STAR and the MCB are not directly applicable to the SUMC processor since in the SUMC both types of operation are performed within one LSI building block, the ALU.

The Test And Repair Processor (TARP) provides timing and synchronization to the modules and monitors their status after each machine cycle. It also initiates reconfiguration and controls the power switches. The TARP does not have any executive functions; normal machine operation is under control of the COP.

VERTICAL PARTITIONING

Vertical partitioning as it is used here is intended to mean partitioning along the data path as opposed to partitioning along the control path which is horizontal or functional partitioning. For this discussion assume that a processor is to be partitioned into two modules. The extention to further degrees of partitioning will be obvious.

The advantage of vertical partitioning is that all sub-partitions are identical. Thus as long as any two of the modules are operational a working processor can be configured. If the processor was functionally partitioned into a CU and an AU a working processor could not be configured if all CU's failed even if three AU's remained intact. It is also more probable that one functional type will fail before the other since there is bound to be some inbalance in the design.

The disadvantage to vertical partitioning is that in order to attain the identical sub-partitions there must be an undesirable repetition of control functions to allow the module to assume either half of the word as well as special controls which identify which half word it is representing at any time, and interface logic to allow communication between the half words. This also complicates the switching and consequently the BOSS functions associated with configuration control. In this configuration the BOSS would have to make twice as many switching decisions, with the incumbent positional determination, to interconnect the processor with the other computer subsystems and the interface between the processor sub-partitions would have to be two directional, doubling its size and the switching decisions at that interface as well.

To consider implementing unique halves, as has been suggested, would be to sacrifice the main advantage of this approach - identical sub-partitions - to alleviate but not eliminate many of the disadvantages. The positional selection and dual switching between processor halves would be eliminated but functional controls would still be duplicated and two switching decisions would still be required to connect the processor to the remaining computer subsystems.

INTERNAL REDUNDANCY

Internal redundancy implies that some or all of the sub-functions within a module class are replicated as spares within a particular module and are not commonly switched between sub-partitions of other modules in that class. The replicated functions effectively enhance the reliability of the module of which they are a part. A common example of this type of partitioning is the addition of spare bit planes within a memory to be switched in upon detection of a failed bit driver. Thus the single failure of a bit driver could not destroy an entire memory module. This technique has also been considered for control memories where the memory technology does not afford a reliability equivalent to that of the logic technology.

This partitioning technique can be used to advantage if one failure mode overwhelmingly degrades the reliability of the whole module or if complete switching capability between sub-partitions becomes unwieldy due to the number of signals or the number of possible combinations (switching decisions). The penalty for this type of partitioning is an increase in gate count and the incumbent increase in size and power requirements.

PARTITIONING OF THE SUMC PROCESSOR

The one obvious conclusion to be drawn from the survey of partitioning techniques is that functional partitioning entails the lease amount of replication of logic. This is undoubtedly the main reason that it is so heavily favored by those engaged in the design of ultrareliable spaceborne computers. The system block diagram of the SUMC processor shown in Figure 5 depicts the highly functional organization of SUMC. The LSI building blocks from which the processor is built are also functional in nature. The first attempt at partitioning SUMC was therefore the functional approach.

Figure 5 also shows that the ALU has a high fan-in as does the FPU which uses an ALU building block as an input multiplexer. These two units also have several common inputs. It was thus determined as an initial guideline that these two units should be contained within the same sub-partition. The

MRU has a large fan-out and interconnects to every other functional unit and so became the pivotal unit in the partitioning scheme.

Figure 6 shows the results of the first partitioning in which the control functions were separated from the arithmetic functions. The card counts below each unit correspond to the SUMC breadboard cards, each of which will be reduced to an LSI part in the final version with the exception of the Instruction Address Read Only Memory (IAROM) and the Program Read Only Memory (PROM) which are already in LSI form. Consequently the partitions are more balanced than is indicated by the gate counts or card counts given. The number of signals crossing the CU/AU interface was extremely high for this partition, however.

A second partition was therefore attempted in which the MRU, which contains the instruction register, was moved into the control unit. The results of this partition are shown in Figure 7. This appears to be a more balanced design but the card and gate counts do not account for the LSI/ROM's.

The figure shows that while the control signals crossing the AU/CU interface have been reduced somewhat, the data paths through the MRU have created more interconnections than they have saved. Table 1 points up the glaring inefficiency of the attempted partitions. Of the three processors surveyed, the MARCS had the highest pin count which with control signals added would approach 200 pins. The MCB which is closest to the ARMMS concept contained only 153 switchable interconnections. The STAR is probably an unfair comparitor in this context because of its low capacity.

RECOMMENDATIONS

Based on these initial attempts at partitioning the SUMC processor, some preliminary recommendations can be made. First, partitioning to more than two sub-partitions should not be attempted. Partitioning to lower levels can reduce unit unreliability and unit pin counts but will always increase the total processor pin count and further complicate the switching problem. Also, the initial reliability analysis indicated that the reliability goal of .99 for 5 years could be achieved with two processor sub-partitions.

Several approaches can be taken to reduce the pin counts of the SUMC sub-partitions. They may be considered individually or in several combinations. The data paths could be reduced to half-words or bytes at all switchable interfaces. The merit of this has to be weighed against the speed requirements as well as the reliability analysis. Table 2 shows the improvement in the pin counts for the two SUMC partitions with the data paths reduced to 8 bits at the switchable interfaces. There would actually be a slight increase in gate count as well but this is negligible. The pin counts are still quite high because of the large number of control signals which were not necked down. It was felt that this would affect the timing and thus represent a major deviation from the SUMC design, the effects of which could not be evaluated in a reasonable time.

The only convenient method of dealing with the large number of control signals which pass betwen the CU and AU short of a major redesign is to use internal redundancy. As can be seen in Table 2, this effectively reduces the number of switchable lines to that of the unpartitioned processor. It also doubles the gate count of the processor. However, this gate count is not out of line with the gate counts of existing processors and so the approach is considered viable. A switching interface would not be necessary between the CU and AU nor would it be necessary to add any logic. The appropriate copies of CU and AU would be connected by power switching under control of the BOSS as is done on the STAR and MARCS computers to connect stand-by spares to the bus system.

If byte interfaces were used with an internally redundant processor the result would be a manageable number of switchable interconnections and a processor which was internally as fast as a non-partitioned processor. Thus if the byte conversion units could be run on a faster clock there would be little or no speed degradation. If in addition, as the preliminary system architecture study suggests, it is not necessary for the processor to communicate directly with the I/O unit further pin savings can be achieved.

The alternative to these techniques, should reliability analysis or architectural considerations prove unfavorable, is a major redesign of SUMC or the incorporation of a different processor into the ARMMS system. The redesign would be aimed at reducing the number of control signals which cross the CU/AU interface. In general terms this would be accomplished by eliminating individual control of all subpartitions in favor of common controls which can be decoded by all subpartitions simultaneously. To a certain degree the AU should also have the capability to react to conditional controls independently. This control philosophy is illustrated in Figure 8.

During Phase 2, additional partitioning alternatives will be examined which may make a redesign unnecessary. One such scheme, suggested by MSFC, would split the PROM vertically and place those bits which communicate directly with the AU, into the AU. Those which require further decoding would remain in the CU.

TABLE 1

COMPARISON OF SUMC PARTITIONS WITH EXISTING PARTITIONS

PARTITION	PINS	GATES
SUMC (A)		
CU	163	752
AU	268	5517
TOTAL	431	6269
SUMC (B)		
CU	248	2656
AU	278	3613
TOTAL	526	6269
MCB		
CU	130	8100
AU	23	3400
TOTAL	153	11, 500
STAR		
COP	18	NOT AVAIL- ABLE
LOP	14	
MAP	14	
ROM	14	
TOTAL	60	
MARCS		
PCU	45	750
ALU	54	1000
CS	63	500
TOTAL	162+	2250

TABLE 2

COMPARISON OF ALTERNATE PARTITIONING TECHNIQUES

PARTITION	PINS	GATES
SUMC (A)		
CU	163	752
AU	268	5517
TOTAL	431	6269
SUMC (B)		
CU	248	2656
AU	278	3613
TOTAL	526	6269
(A) BYTE INTERFACE		
CU	149	752
AU	172	5517
TOTAL	321	6269
(B) BYTE INTERFACE		
CU	154	2656
AU	147	3613
TOTAL	301	6269
INTERNAL REDUNDANT	109	12,538
I.R. & BYTE	37	12,538
NO I/O	29	12,538

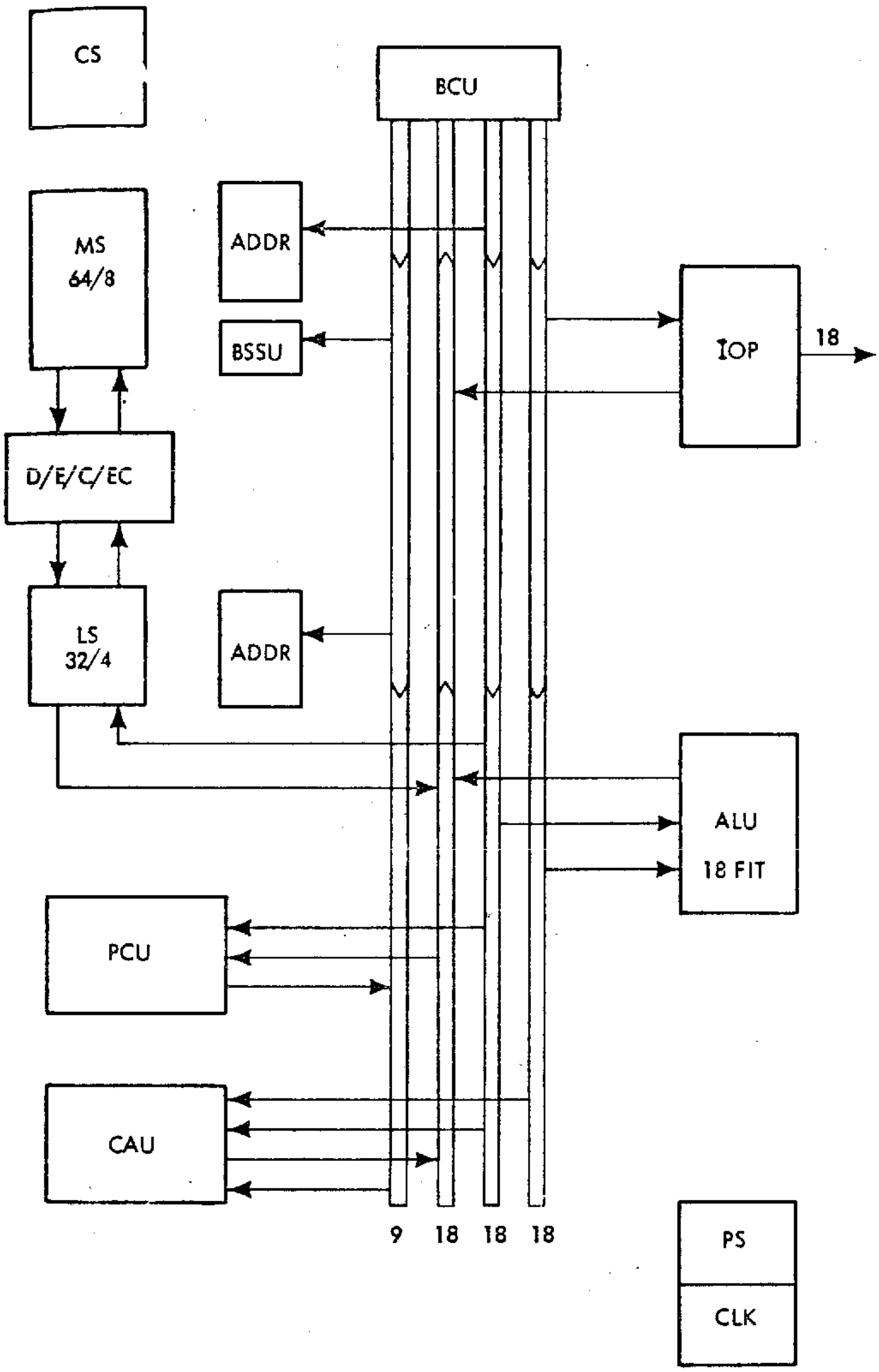


FIGURE 1. BLOCK DIAGRAM OF THE MARCS COMPUTER

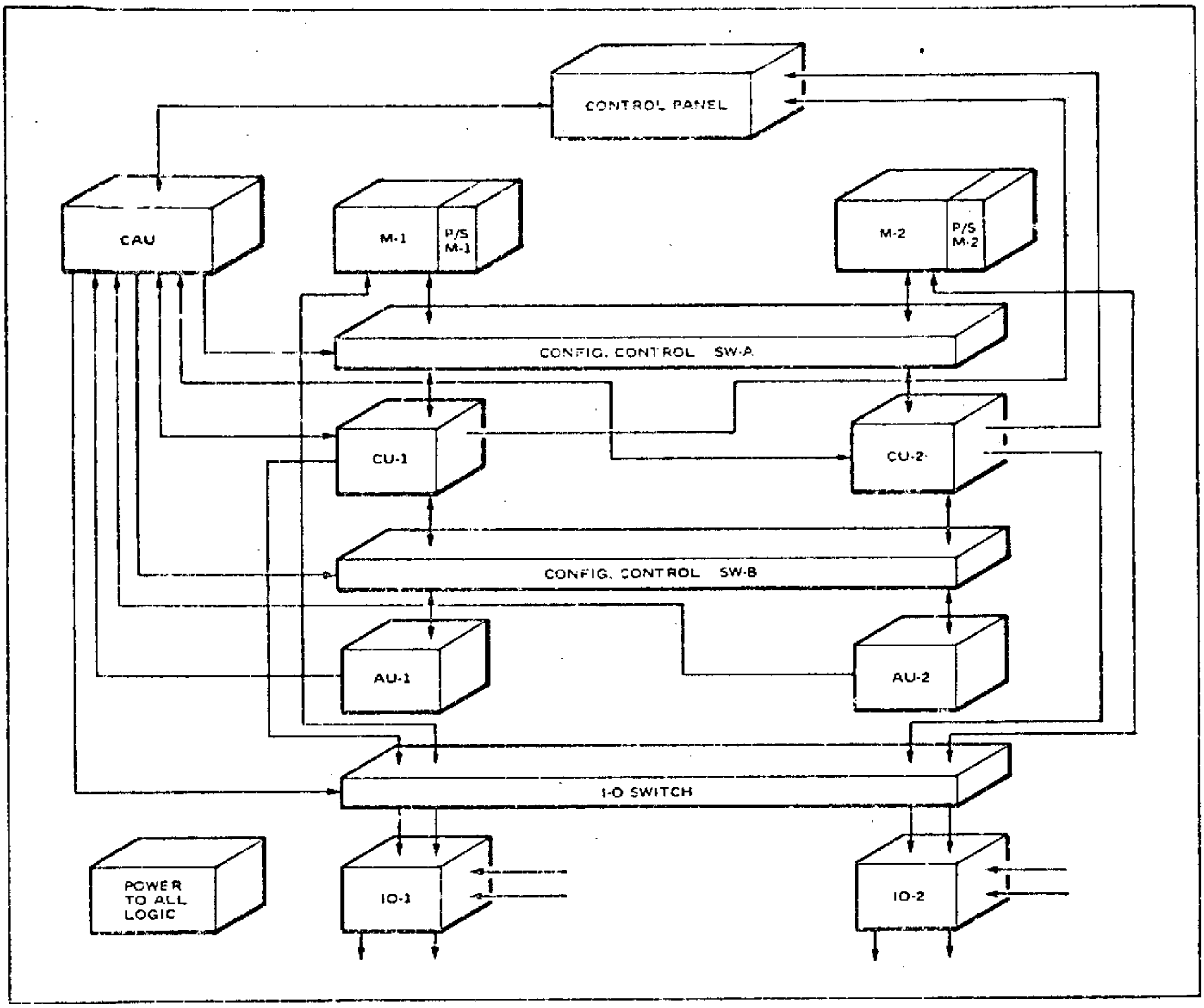


FIGURE 2. BLOCK DIAGRAM OF THE NASA-MCB

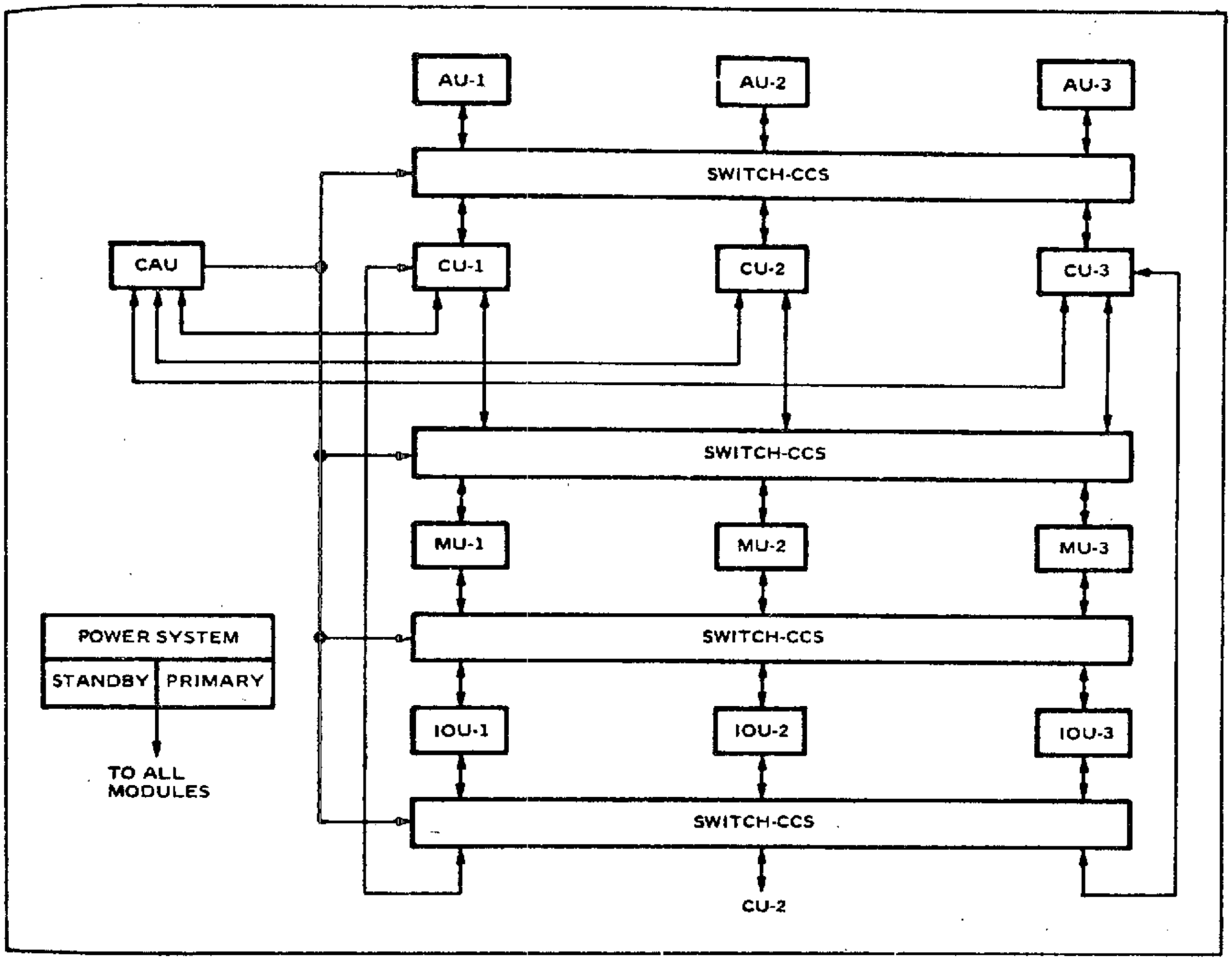
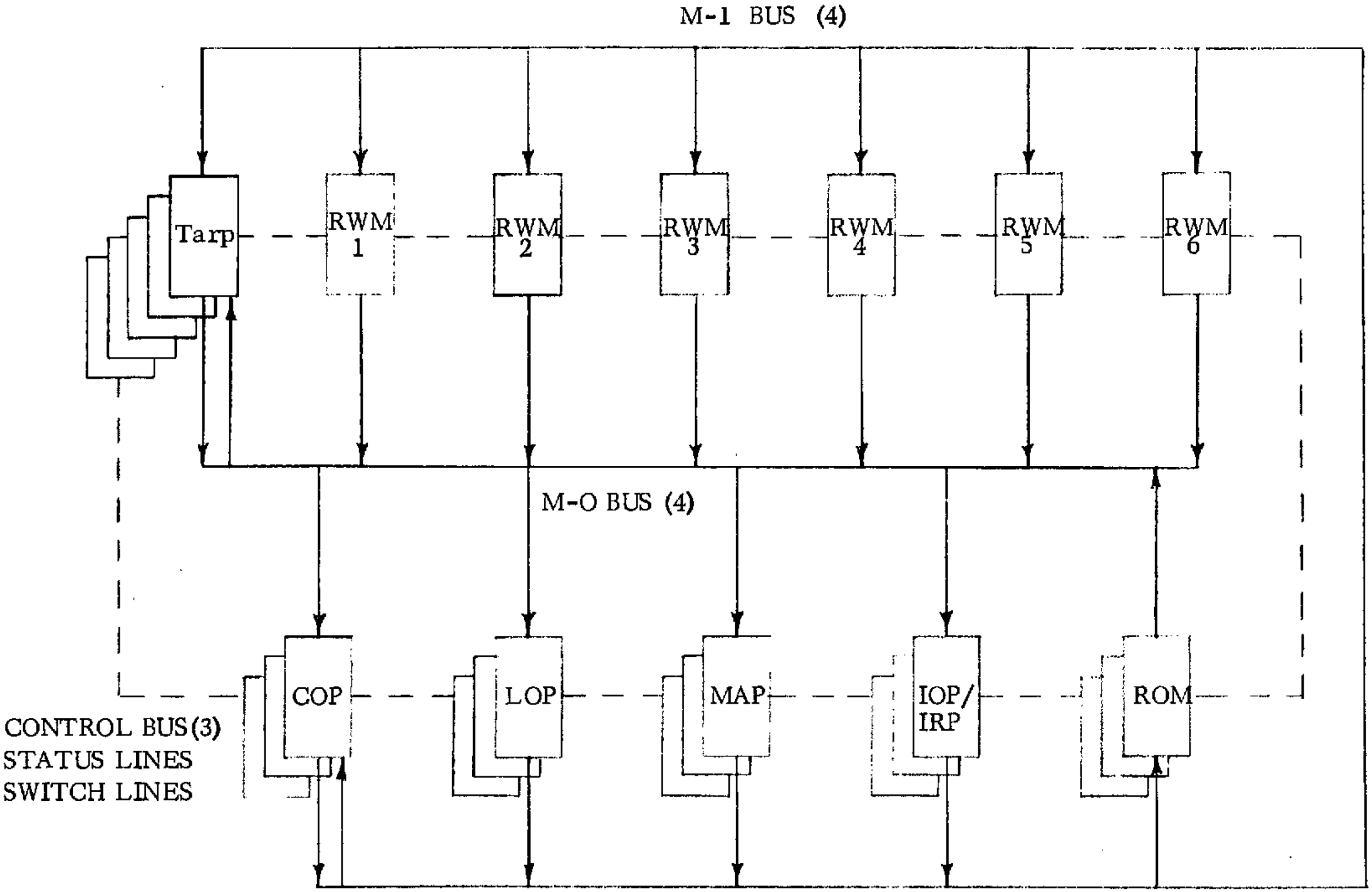


FIGURE 3. EXPANSION OF THE NASA MODULAR COMPUTER



BLOCK DIAGRAM OF THE JPL-STAR COMPUTER
FIGURE 4.

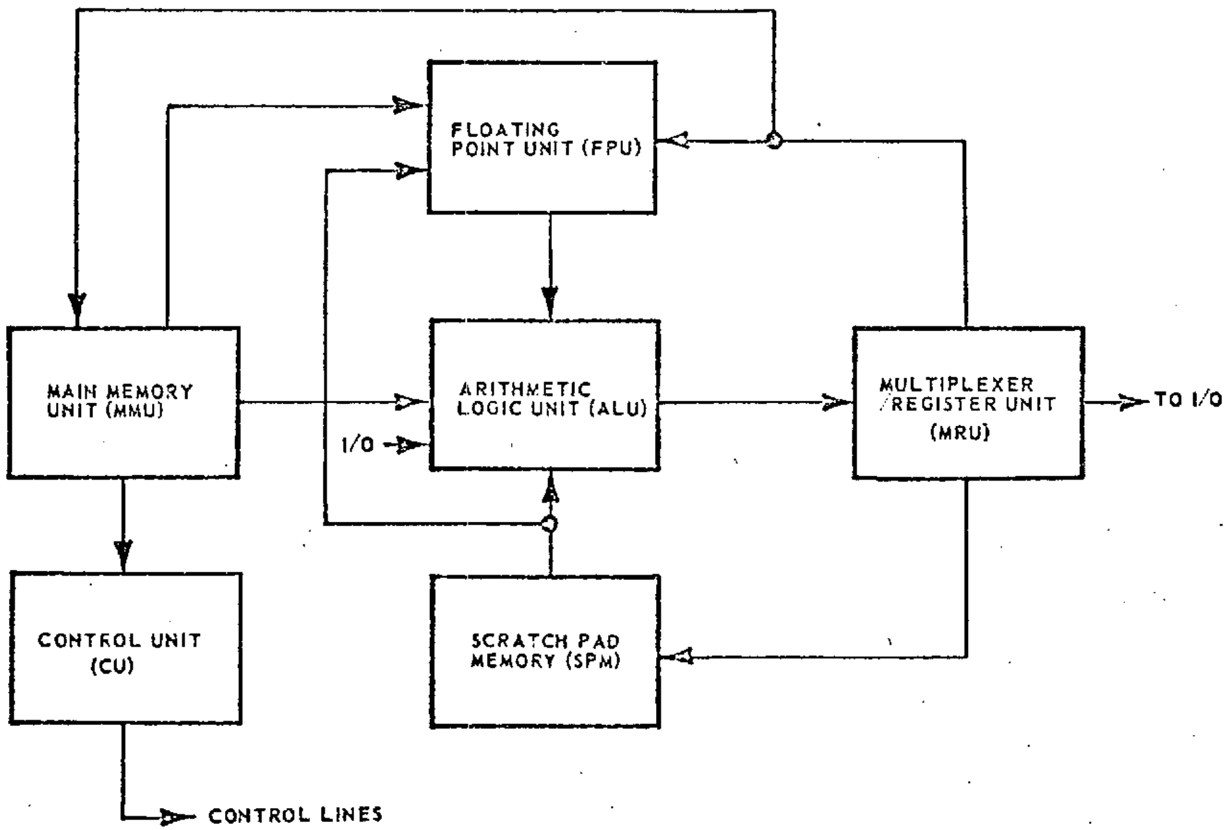
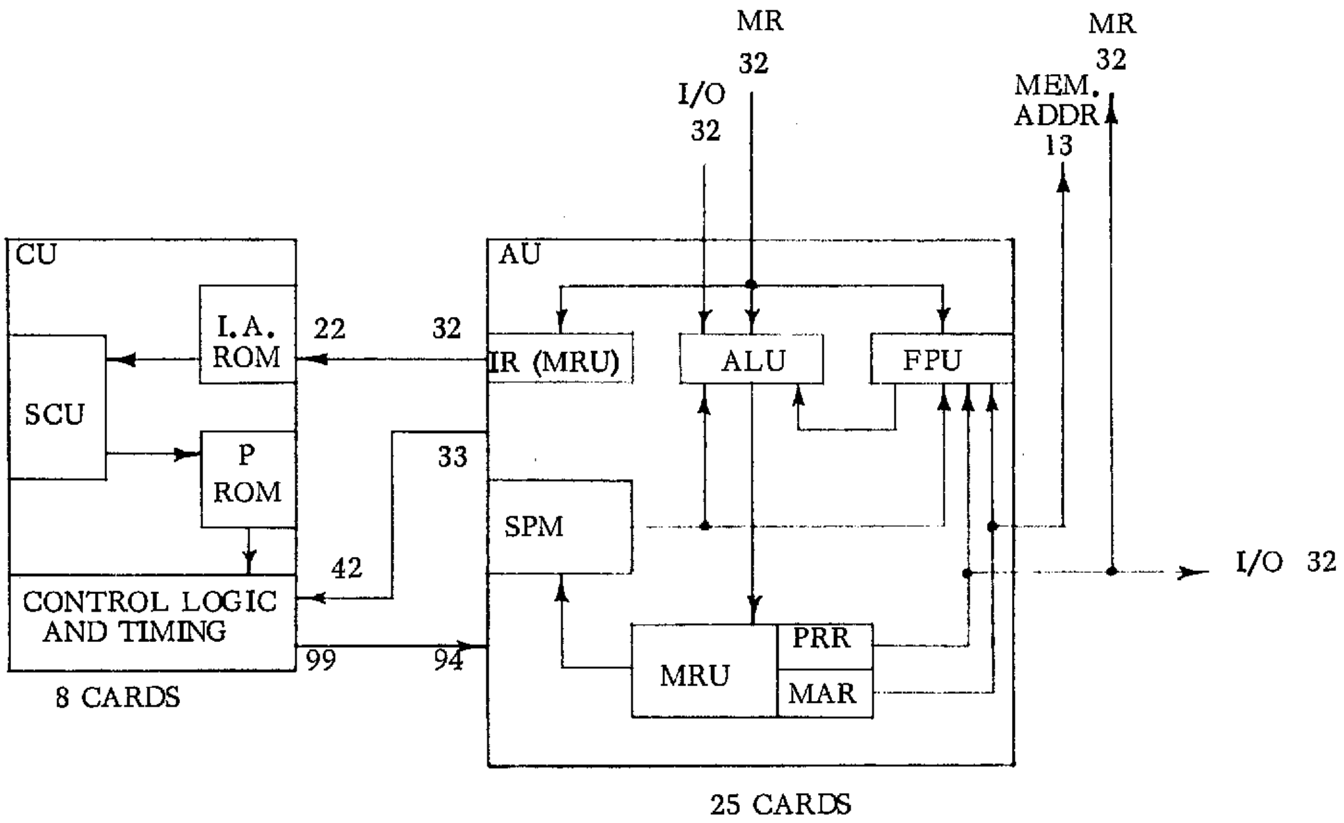
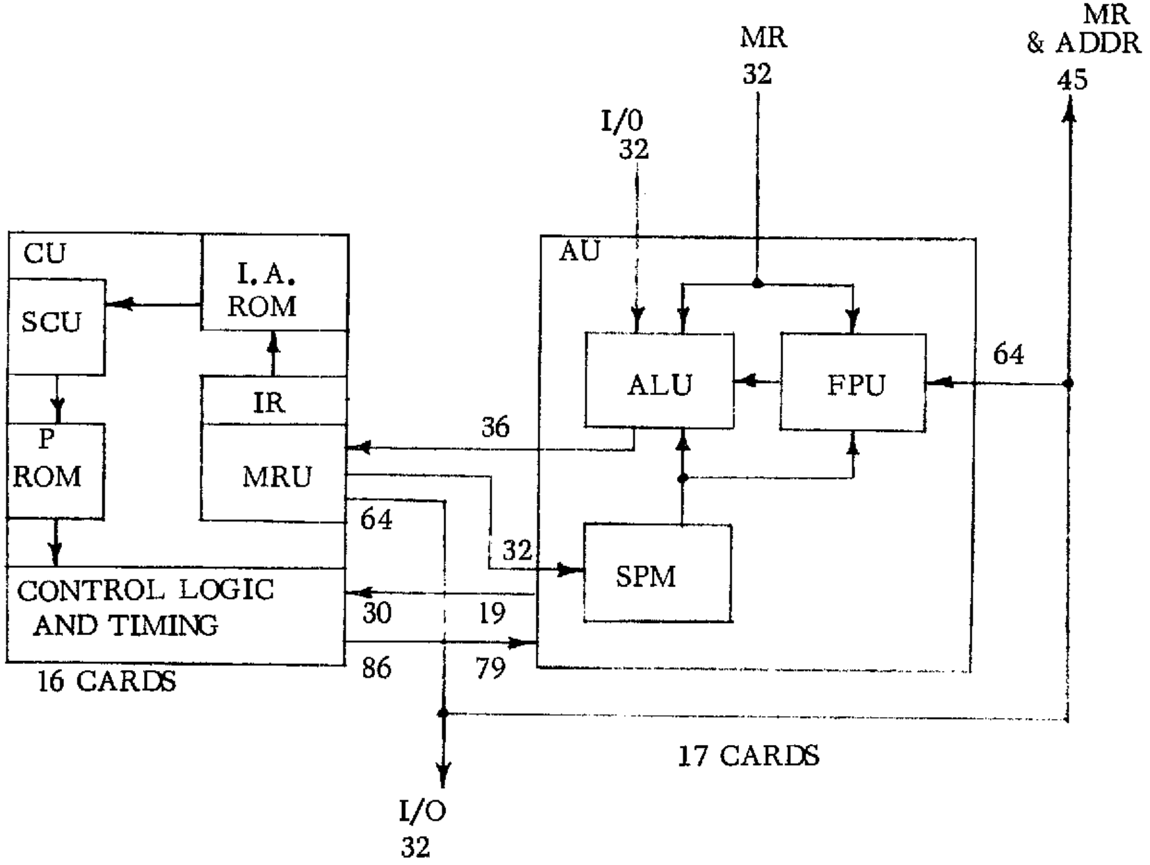


FIGURE 5. SYSTEM BLOCK DIAGRAM OF THE SUMC PROCESSOR



SUMC PROCESSOR PARTITION A

FIGURE 6.



SUMC PROCESSOR PARTITION B

FIGURE 7.

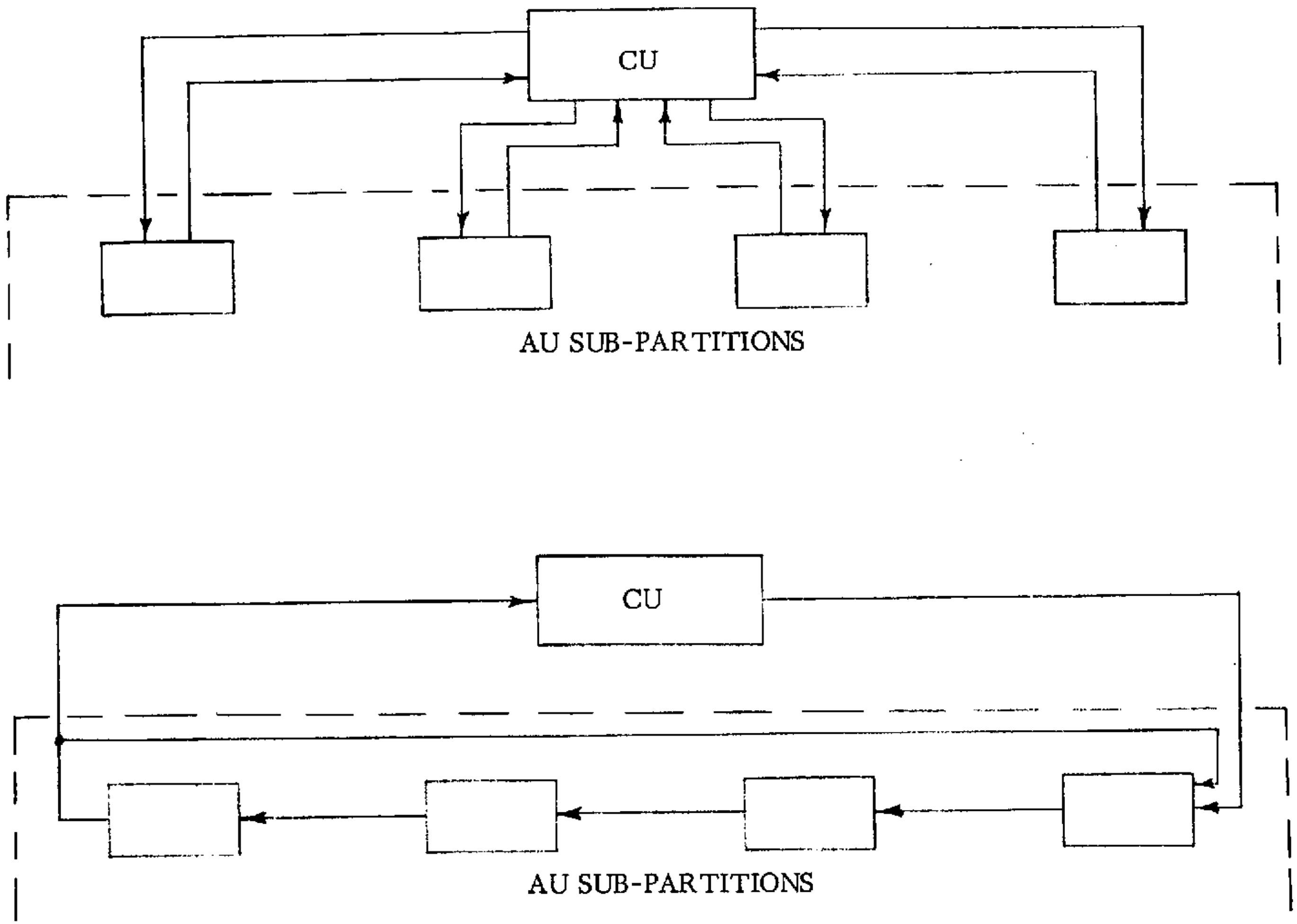


FIGURE 8. REDESIGN FOR A CONTROL PHILOSOPHY AMENABLE TO PARTITIONING

SECTION 8

DATA TRANSMISSION STUDY

A potentially limiting aspect of the ARMMS architecture is the achievement of data transmission rates among the modules at a sufficient rate to allow achievement of the design objectives of both redundant and high computing capacity modes. Interface design factors which may prove to be formidable obstacles include pin count, power required by interface circuits, and speed.

In order to gain better preliminary visibility of the electrical problems involved, a preliminary study of the feasible approaches to implementing the transmission lines between ARMMS modules was conducted. This included a survey of presently available driver and receiver circuits, a review of possible module interconnection approaches, and consideration of the possible physical constraints (weight, power, etc.) which may be required for ARMMS.

The basic conclusion reached was that 1) transmission speeds of from 15-20 MHz using TTL circuits are feasible within reasonable power levels and interconnection techniques; 2) differential line receivers are preferred for noise rejection but impose a penalty in pin count; 3) etched contour cable is an attractive intermodule connection technique; 4) the upper limit of number of modules suggested by baseline 1 does not impose basic difficulties on the design of the transmission system; 5) ECL circuits have significant disadvantages for use in a bus-oriented transmission system in spite of their attractive speed-power characteristic.

The implementation of the transmission paths will be considered in more detail during Phase II, but the initial study has allowed transmission speeds of more than a few megahertz to be used with confidence in further refining the design.

DATA TRANSMISSION

Summary

This report discusses party line data transmission systems with emphasis on hardware, speed and performance characteristics. For data transmission at speeds less than 50 Mbps, current source drivers are optimum for line interfacing. At higher speeds, voltage source drivers are required. Differential line receivers are preferred over single-ended receivers due to their common mode rejection capability. Presently available bipolar integrated circuits can manipulate data at rates up to 15 megabits, with the exception of the ECL families which can manipulate data at 100 and 200 megabits. With regard to the data link, a transmission line with uniform characteristic impedance and with short stubs for interconnecting the modules is recommended.

1.0 INTRODUCTION

The maximum usable speed and performance of a party line data transmission system is a function of several parameters including the noise environment, the driving and receiving method, the transmission line and its termination technique, the stub lengths, and the choice of connectors. This report discusses these factors, all of which affect the fidelity of data transmission, and establishes some performance bounds.

This report separates the transmission system in two fields. The first is the interface area including the types of drivers and receivers available, and the parameters essential in the definition of their performance. A summary of the presently available solid state hardware as a function of type, speed, and power is presented. The second field investigated is the transmission medium. Various types of transmission lines are discussed from a point of view of performance within the system. In addition, rules are given for securing maximum EMI protection, and for minimizing distortions due to stubs, interconnect and terminations.

This report makes use of the data and experience that was acquired in a party-line data transmission study of an airborne multiplexing system using 300 feet of cable with 32, 20-foot stubs, and bandwidths extending to 4 MHz. In addition, this report makes use of experience gained on high-speed processors such as the 15 megabit per second PCM encoder included in the Multispectral Scanner on board the Earth Resources Technology Satellite (ERTS).

2.0 INTERFACE CONSIDERATIONS

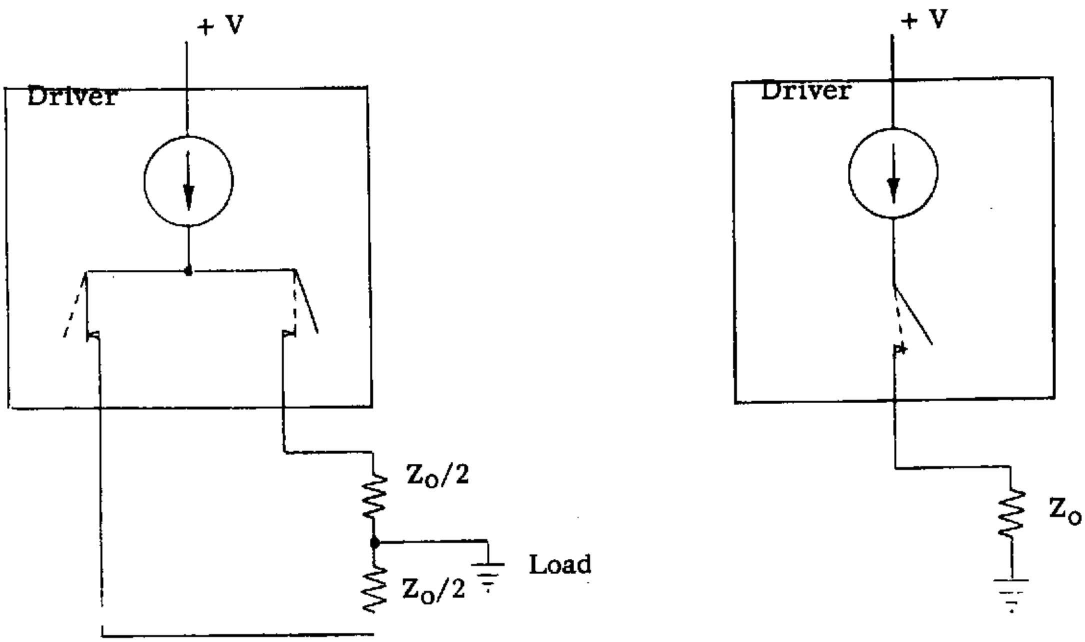
Two aspects of the interface problem will be addressed; first the types and performance of line drivers and line receivers, and second the specific capabilities of presently available devices:

2.1 Interface Circuit Types and Performance

2.1.1 Line Drivers. Line drivers can be either the high impedance current source type, or the low impedance voltage source type. Each of the two types has its advantages and limitations which make selection a function of the application.

2.1.1.1 Current Source Drivers. Current source drivers are high output impedance devices which divert a fixed amount of current into a load. The current can be differential as shown in Figure 1a where current is diverted into each half of the output load. It can also be single ended as shown in Figure 1b where current is pumped into the load only when the data is in the logical one state. It is shown in Appendix A4 that the differential driver is electrically equivalent to the single ended driver.

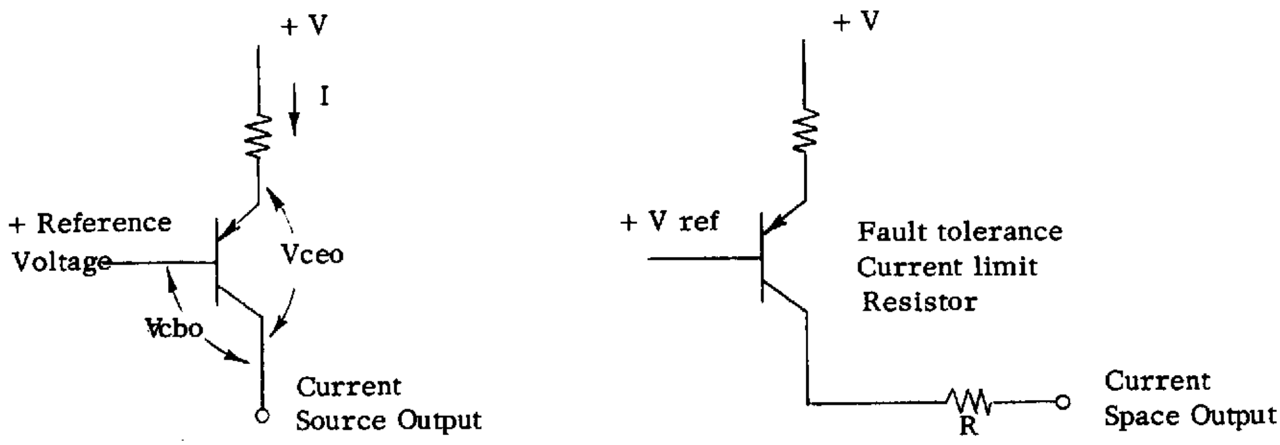
The high output impedance property of the current source driver allows for three important features ideal for a party-line system. These include line isolation, driver fault tolerance, and high common-mode output voltage range. Line isolation is automatically provided by the high impedance output stage of the line driver, which in most devices is the collector of a bipolar transistor. Line isolation yields non-distortion of reflected waves. This characteristic simplifies the modeling of the transmission line, and in some cases, can allow reflections to occur on the line without degrading performance. Therefore, settling of the waveform can occur as



a) Differential Driver

b) Single Ended Driver

FIG. 1 Current Source Line Drivers



a) When current source output is tied to a negative power supply, automatic fault tolerance is provided by the collector to Base and Collector Emitter Breakdown voltages.

b) Resistor R provides current limitation in the event of an output short to a positive power supply, or to a negative power supply of voltage greater than the Vcbo or Vceo.

FIG. 2 Current Source Driver Fault Tolerance Characteristics predicted by analysis (this point is further discussed later in this report). Fault tolerance against the transmission line shorting to a power supply voltage during system test is a desirable feature. It is automatically provided by a sufficiently high collector-base breakdown voltage of the output transistor as shown in Figure 2a (for a voltage higher than the collector base breakdown voltage, there is no protection). For additional circuit isolation, a current limit resistor can be economically added. This resistor will not alter the system performance, however it will protect the driver against a system test accidental short of the data link to a high voltage supply of either polarity. Besides system test protection, the current limit resistors provide an economical insurance against a line driver failure affecting the data link, and thereby the transmission between other modules. The line driver failure can be of any form such as a short to ground, to power supply, or other, so long as the failure is passive, (i.e. the driver does not transmit data when not enabled). Due to the high output impedance of current source drivers, the output voltage developed across the load is independent of the common-mode voltage at the load. Thus, these drivers have the characteristic of withstanding high common-mode output voltage range. This property is important since a common-mode voltage can be applied on the line when data is present in order to identify presence of data. More on this point will be explained in Appendix A4.

There are currently available integrated circuits line drivers with current sources extending from 3 to 12 milliamperes, and capable of accommodating data rates up to 15 Mbps. Higher switching speeds in integrated circuits are expected to be available; however, discrete types of current source drivers can be designed at present with switching speeds in the order of 100 to 150 MHz. The limiting factors involved are the components selection, the circuit layout, and the external capacitance other than line capacitance, since the line appears resistive with characteristic impedance Z_o. The following example illustrates the effect of capacitance on the maximum operating frequency.

<u>Example</u>: Given an ideal current source I with switching rise time less than one nanosecond, find the maximum operating frequency when the driver is connected to a properly terminated 50 ohm line through a connector with 50 pf of capacity.

<u>Solution</u>: The setup for problem is as shown in Figure 3a. The equivalent model for the analysis is shown in Figure 3b.

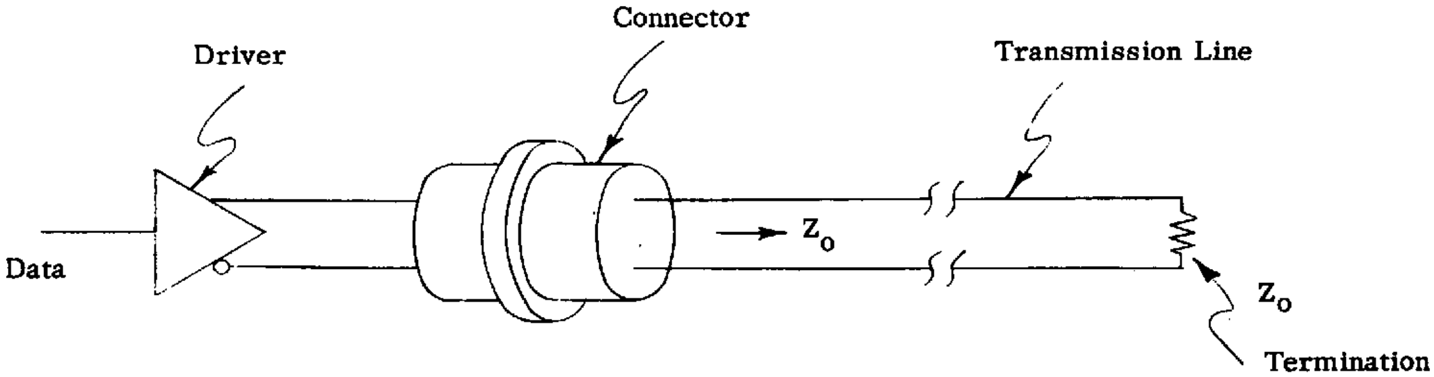
The solution can easily be derived to be:

$$V_c\ (t) = IZ_o\ (1 - e^{-t/Z_oC})$$

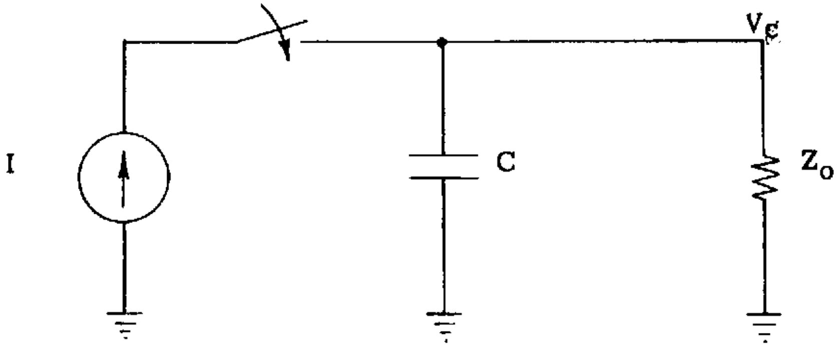
Three times constants yield $t = 3 \times Z_oC = 3 \times 50 \times 50 \times 10^{-12} = 7.5$ nsec.

where t = rise time

A bit time of twice the rise time is then $2 \times t = 15$ nsec, and the maximum bit frequency is $1/15 \times 10^{-9} = 67$ MHz. Therefore, to transmit high frequency data with a current source, it is very essential to minimize the external capacitance.



a) Physical problem configuration



$$V_o(S) = \frac{I}{S}\left(\frac{Z_o/sc}{1/sc + Z_o}\right)$$

$$V_c(f) = IZ_o\left(1\text{-}e^{-t/Z_oC}\right)$$

b) Analytical model

FIG. 3 Physical and analytical model for sample problem

2.1.1.2 Voltage Source Drivers. Voltage source drivers are low output impedance devices, and in practice, with proper design, can drive capacitive loads at frequencies up to 250 MHz. Unlike the current source drivers, the voltage source drivers have three distinct disadvantages which make adaptability to party line operation difficult. First, the low output impedance of the device restricts the use of line interface current limit resistors to achieve isolation from the data link. Second, voltage drivers have low common-mode voltage range since line voltages affect the driver performance. In most commercially available drivers, diode clamping to power supply and ground is provided to short out any common-mode voltage. If the common-mode voltage has a low impedance source, damage to the driver can occur. A third but not as serious disadvantage is the difficulty of obtaining high output impedance in the standby mode. Unlike the current source driver, this is not a natural feature of the voltage driver and requires additional circuitry.

2.1.2 Line Receivers. Line receivers are high impedance devices with either single ended or differential inputs. Before discussing the difference between the two types, a discussion of the most important parameters defining line receiver performance is given. These parameters include the following:

a. Common Mode Rejection Ratio (CMRR) - The CMRR is the ratio of open-loop (differential) gain to common-mode gain. The CMRR falls off at high frequencies mainly because of internal capacitance effects. Since the capacitance of the input stage has more effect with high source impedance, the rejection ratio is also a function of the source impedance. Figure 4 shows the dependence of CMRR upon frequency and source impedance. CMRR is an important parameter for a line receiver since high frequency common-mode voltage can be coupled onto the data link from external sources and can appear as a differential signal if the rejection is poor.

b. Common-Mode Voltage Range (CMVR) - The common-mode voltage range is defined as that voltage applied simultaneously to both input terminals which, if exceeded does not allow normal operation of the receiver. Thus, this parameter is important since the line may purposely operate at a common-mode voltage other than ground in order to automatically shut off a differential bias. This last point is explained in detail in Appendix A4.

c. Input Impedance - Input impedance of the receiver defines the coefficient of reflection (and thereby the voltage attenuation) at every interface with the line. A simple technique to compute the received voltage at the furthest receiver is to consider the final-value voltage on the line when the driver is loaded with all receivers in parallel.

   Example: Consider a 6 milliampere current source driver driving a lossless line with ten receivers, each with an input impedance of 1.8K ohms. The line is terminated with 100 ohm resistors at each end. The net impedance is then 10, 1.8K ohm in parallel with two 100 ohms, or 39 ohms. The final value output voltage on the line is then 6 ma X 39 ohm = 234 mV.

d. Input Sensitivity - The input sensitivity is defined as the differential dc voltage required at the inputs of the receiver to force the output from one digital state to the other. This parameter defines the minimum peak-to-peak signal level required for operation with a signal-to-noise ratio of zero dB. The input sensitivity of presently available line receivers varies from 2 mV to 1000 mV.

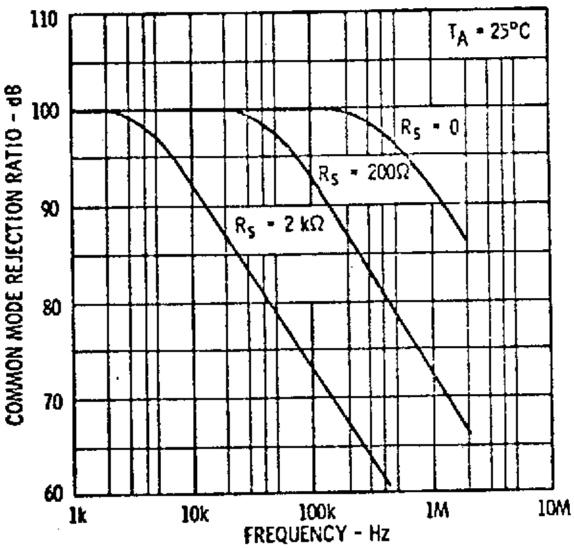


Fig. 4 Common Mode Rejection Ratio as a Function of Frequency.

e. <u>Offset Voltage</u> - Offset voltage is that voltage which must be applied between the input terminals through two equal resistors to force the output to its midpoint (between a logic 0 and a logic 1). The offset voltage is basically then a bias error on the input, and must be accounted for in determining the worst case allowable input voltage.

Differential line receivers are generally preferred/single-ended receivers because of their high common mode rejection and high common mode range. Single-ended receivers are simpler in design; however, for a data link application differential line receivers are recommended. The described parameters not only define a receiver performance, but also can be used to calculate the required driver differential output voltage. The example below illustrates the concept. The example assumes some receiver parameters and a system environment. The differential receiver parameters are as follows:

- CMRR: 80 db at 1 KHz; 20 db at 10 MHz
- CMVR: $\pm$ 3 V
- Input impedance: 1.8K ohm
- Input sensitivity: 10 mV
- Offset voltage: 10 mV

The system environment is as follows:

- Transmission line losses: 5 db
- Load losses (due to other receivers): 5 db
- Common-mode noise: 1000 mVrms from 1 KHz to 10 MHz
- Transmission bandwidth: 10 MHz
- Coupled differential noise in band of interest: 10 mV RMS

To maintain a signal-to-noise ratio of 20 db, the problem is to find the differential driver output voltage when the line is terminated in its characteristic impedance at both ends.

<u>Solution</u>:

RMS Signal to overcome the noise (0 dB SNR) = 10 mV RMS
RMS Signal for a SNR 20 db = 100 mV RMS

10 db loss compensation (5 dB transmission loss plus 5 dB load loss)	316.2 mVRMS
Additional signal for common-mode compensation (1000 mVRMS/20 dB = 100 mV)	100.0 mVRMS
Total RMS signal	416.2 mVRMS
Peak-to-peak (differential) signal (twice RMS signal)	832.4 mV pp
Additional signal for sensitivity compensation	10.0 mV
Additional signal for offset voltage compensation	10.0 mV
Total differential voltage required	852.4 mV

2.2 Interface Devices

A list of the presently available bipolar integrated circuits line drivers and line receivers is given in Tables I and II. These tables show that the TTL compatible line drivers and receivers can be operated at a maximum frequency of fifteen megabits. Higher speed operation (15 to 30 MHz) can presently be achieved with discrete line drivers, and with a combination of discrete and IC components for the line receivers. At very high speed (100 to 200 MHz), use of ECL logic is strongly recommended. The ECL family provides IC line drivers and line receivers for operation at these speeds.

TABLE I SAMPLE DATA OF PRESENTLY AVAILABLE LINE DRIVERS

Line Driver	Manufacturer	Max. Operating Speed (MH_z)	Power Dissipation Per Driver (non Loaded) On (MW)	Power Dissipation Per Driver (non Loaded) Standby (MW)	Power Delivered to 50 ohm Load (MW)	Differential Line Voltage (MV)	Power Supply Dissipation with 50 ohm Load (MW)
DM 7830	National	20	60	-	140	2600	330
SN 55109	Texas Instruments	10	70	70	1.8	300	100
SN 55110		10	95	95	3.6	600	155
RAI 245	Radiation	15 MH_z	25	1	.45	150	25
MECL 10000 Series Gates	Motorola	100	35	-	16	900	125
MECL III Series Gates	Motorola	200	70	-	16	900	160

TABLE II SAMPLE DATA OF PRESENTLY AVAILABLE LINE RECEIVERS

Line Receiver	Manufacturer	Max Operating Speed (MH_z)	Power Dissipation Per Receiver	Sensitivity (mV)	Common Mode Voltage Range	Comments
DM 7820A	National	20	10 18 25	60-1000 60-500 60-1000	+15V 0 -15V	Device has excellent CMVR, however the sensitivity specs make this device unusable with low voltage output line drivers.
SN 55107	Texas Instruments	10	130	5 *	+3V -3V	Sensitivity limits not specified
RAI 248	Radiation	15	20	75 *	700 MV *	Sensitivity and CMVR not specified
MC 10115	Motorola	100	25	*	*	New product, data sheet not available.
MC 1692	Motorola	200	55	600	*	CMVR not specified.

3.0 TRANSMISSION CONSIDERATIONS

A transmission line is considered here as the transmission medium between modules instead of fiber optic line or other types of links. The performance of a data link system is a function of the transmission line noise rejection properties, its losses, and its loads (including stubs), in addition to the receiver properties with respect to detection. Optimizing the performance of a data link system means minimizing the probability of error for a given signal level and environmental noise, and a given transmitter and receiver configuration. Since the intent of this system is to maintain simplicity in the design of the receivers (i.e. simple detection), the burden of proof to minimize phase and amplitude distortions is placed on the data link. In other words, it is essential to first select a transmission line that has good rejection to Electro-Magnetic Interference (EMI), and second, to connect this line such that rejections due to the various loading elements are kept to a minimum.

This report discusses the transmission line from these two standpoints, first from the hardware aspect in due consideration to EMI; and second, from a geometry standpoint with guidelines to minimize distortions.

3.1 Transmission Lines

Various types of transmission lines are available for the transmission of data. The selection of a line for an application is a function of the line length, line characteristics, line loss, impedance, transmission bandwidth, and noise environment. Line length can be translated to the time domain through the propagation time delay. That delay ranges from 1.5 to 2.0 nanoseconds per foot of cable. Thus, a four foot cable is six to eight nanoseconds long. A cable is generally considered "long" if its length in nanoseconds is of the same order or greater than the signal rise time.

3.1.1 Open Wire and Wire Over Ground Plane. Open wire cables have an uncontrolled impedance with wide variations in characteristic impedance ranging from 50 to 300 ohms. Wire over ground plane has a more controlled impedance than open wire. Both of these wiring techniques have the singular disadvantage of being very susceptible to EMI coupling. This is because single wire transmission has no shielding to attenuate either high frequency, high impedance magnetic fields or high impedance electrostatic fields. (Please refer to Appendix 3 for more details on methods to minimize EMI coupling.) Single wire transmission can be used provided the signal-to-noise requirements are satisfied. This last requirement is a difficult task to accomplish when the cable is long and the environment is not well defined.

3.1.2 Coax Cable. Coax cable offers many advantages for distributing high frequency signals. The well defined and uniform characteristic impedance of the line permits easy matching. The ground shield of the cable minimizes coupling of high impedance electrostatic interference, and high impedance high frequency magnetic fields. Coaxiable cable though offers little protection to low impedance, low frequencies (less than 5 KHz) magnetic fields derived for example from a-c power sources. This last disadvantage may not be significant, depending upon the application. The low attenuation at high frequencies makes good coaxial very desirable for the fast rise times (1 to 3 nanoseconds) associated with very high frequency signals (order of 100 MHz). At these frequencies, skin effect is a primary cause of attenuation and coax cable is the only choice for data transmission. The choice between cable size and bandwidth is a compromise that needs to be analyzed at the time of the design.

3.1.3 Twisted Pair Cable. Twisted pair lines, differentially driven into a differential type receiver provide maximum noise immunity. This is because any noise coupled into a twisted pair generally appears equally on both wires (common mode). Twisted pair wise offer excellent rejection to low frequency magnetic fields (20 db) because of the equal and opposite self-cancelling currents induced in the cable. This protection though against magnetic fields coupling is only satisfied when the length of the twist is much less than the wavelength of the interfering signals.

Twisted pair cables have little protection (other than common mode) against high impedance electrostatic fields if the line is terminated in an impedance isolated from ground. If, however, the low impedance terminations are balanced about ground and the cable length is less than .15 of the shortest wave length (see EMI rejection in the Appendix), a high degree of electrostatic rejection is accomplished.

3.1.4 Shielded Twisted Pair. Shielded twisted pair combines the electrostatic shielding advantages of the external shield with the magnetic field rejection advantages of the twisted pair. This wire is almost the optimum with regard to EMI rejection (with the exception of double or triple shielded twisted pair cable). The disadvantages of shielded twisted pair is the greater harness size and difficulty in handling the cable from a product design standpoint.

3.1.5 Ribbon Cable. Ribbon cable is a flat flexible cable with well defined characteristic impedance. Crosstalk is minimized because every other wire is grounded as shown in Figure 5. The advantages of ribbon cable include easy handling of connectors because of the in-line arrangement of wires. EMI rejection from external sources is not as optimum as coax cable because a great percentage of the signal wire is exposed, or non-shielded.

3.1.6 Microstrip Lines. A microstrip line (Figure 6) is a strip conductor (signal line) separated from a ground plane by a dielectric. If the thickness, width of the line, and the distance from the ground plane are controlled, the line will exhibit a predictable characteristic impedance. From an EMI standpoint, the rejection is not as high as coax cable because there is no shielding around the signal line. Partial shielding can be accomplished if, on each side of the signal line, a strip dedicated for grounding is inserted.

3.1.7 Strip Line (Contour Cable). A strip line (Figure 7) consists of a copper ribbon centered in a dielectric medium between two conducting planes. If the thickness and width of the line, the dielectric constant of the medium, and the distance between the ground planes are all controlled, the line will exhibit a characteristic impedance that can be held constant within 5%. The strip line approaches in performance a coax cable from an EMI standpoint if ground strips on both sides of the signal lines are inserted. This is because the signal line has then a surrounding shield.

3.2 System Geometry

The most significant discontinuity in the data links are the stubs used to transfer information from the modules to the transmission lines. These stubs can cause severe reflections, and certain ground rules should be followed to minimize the distortions. Because several stubs are connected to the data link, a detailed examination of the effects of the stubs is given.

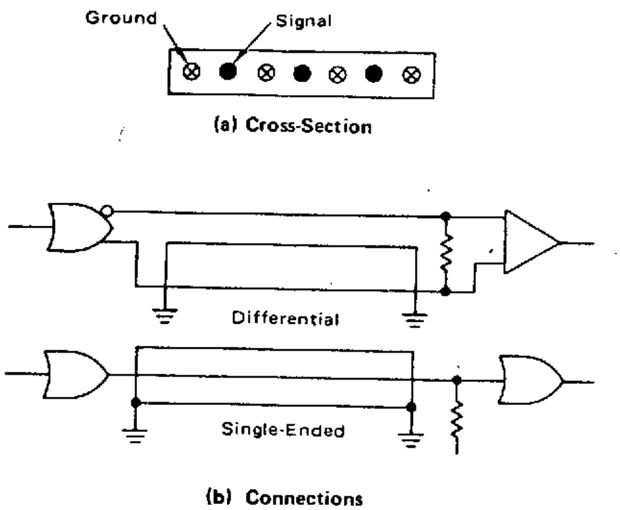


Fig. 5 **Ribbon Cable Interconnects**

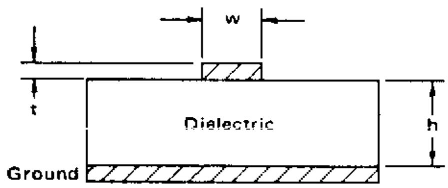


Fig. 6 **Microstrip**

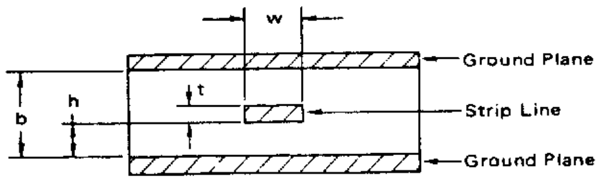


Fig. 7 **Stripline**

3.2.1 Effect of One Stub. Consider a transmission line (see Figure 8) with one stub of length ℓ, one driver directly connected to the line (E), and three receivers two directly connected to the line on either side of the stub, (B, C) and one at the end of the stub (D). The coefficient of reflection at the stub is calculated to be

$$\frac{Z_L - Z_o}{Z_L + Z_o} = \frac{Z_{o/2} - Z_o}{Z_{o/2} + Z_o} = -\frac{1}{3}$$

where Z_L is two Z_o lines in parallel, or $Z_L = Z_{o/2}$. Therefore, as a wave encounters the stub, one-third of its voltage will be reflected back to the source. Two-thirds of its voltage will be transmitted to the rest of the line and to the stub line. The reflected waveforms can be detrimental to the line.

Figure 8 shows the resulting lattice diagram. Figure 9 illustrates the waveforms at points on either side of the stub. Due to the reflections, the system has become more susceptible to noise. Figure 10 shows photographs taken from receivers located on the line on either side of the stub. These photographs agree with the theory.

Figure 11 represents the lattice diagram for the stub. The voltage at the receiver initially exceeds the final value. Figure 12 is a photograph of the waveform at the receiver.

3.2.2 Effect of Multiple Stubs. The inclusion of several stubs on the line greatly disturbs the line due to the multiple reflections. This effect was tested using 300 foot RG108A/U twisted shield pair cable and 28 20-foot stubs connected along the line as shown in Figure 13. Pseudorandom Manchester coded data was transmitted on the data link at a rate of 1 megabit/second. The rise time of the transmitted waveform was limited to 100 nanoseconds. To maintain a good ground, the cable was laid over an aluminum sheet with the cable shield tied to the aluminum sheet at every termination (i.e., at every stub connection to the data link).

Figure 14 is a signal waveform and eye pattern observed at a receiver along the line. This waveform shows the excessive ringing and distortion caused by the discontinuities on the data link. In this application, discontinuities on the cable, resulting from unmatched terminations, imposed echo interference on the line. There are two types of discontinuities which are a source of echoes affecting the high frequency components of the signals. The first echo contributors are the reflections of the signals along the stubs connected to the main line. The stubs are tied directly to the main transmission line at one end, and to a high impedance termination at the other end. When a wavefront on the main line intercepts a stub, a portion of its energy enters the stub, another portion continues its travel on the main line, and a third portion is reflected back to the source. The energy that entered the stub is not dissipated by the stub termination because the termination is a high impedance. It is therefore reflected back to the main line at some later time, thereby introducing an echo. For the low frequency signals, the time delays are small compared to the period of the signal and, therefore, the echoes are approximately in phase with the original wavefront. This is not the case, however, for the high frequency signals, where the echoes are additive at the passage of each stub and can be significantly shifted in phase from the original signal.

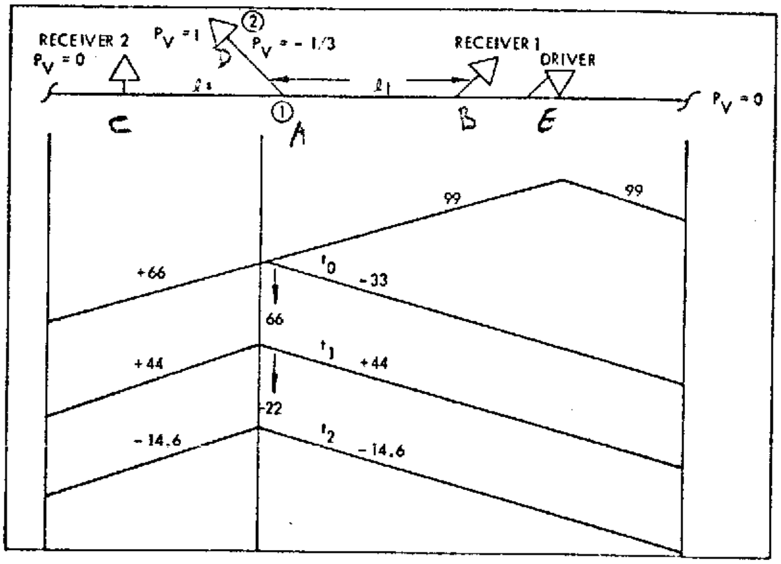
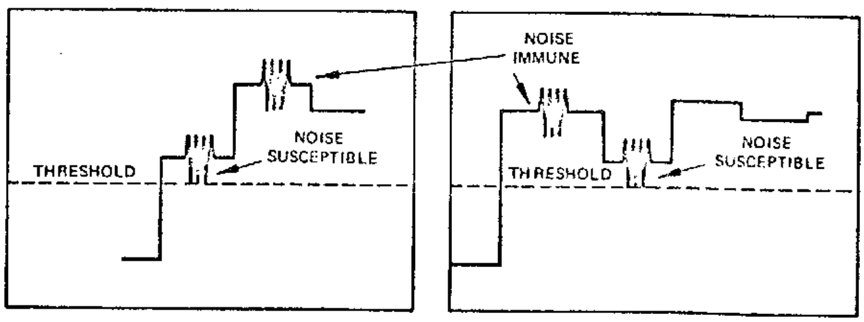


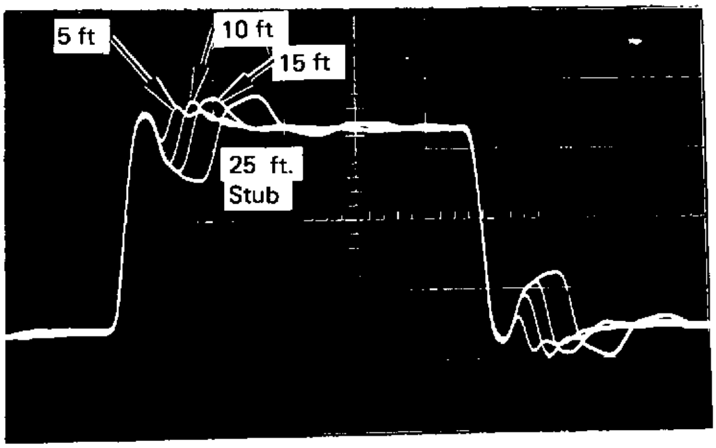
Figure 8. Lattice Diagram Showing One Stub Tied to Line



a. Receiver 2

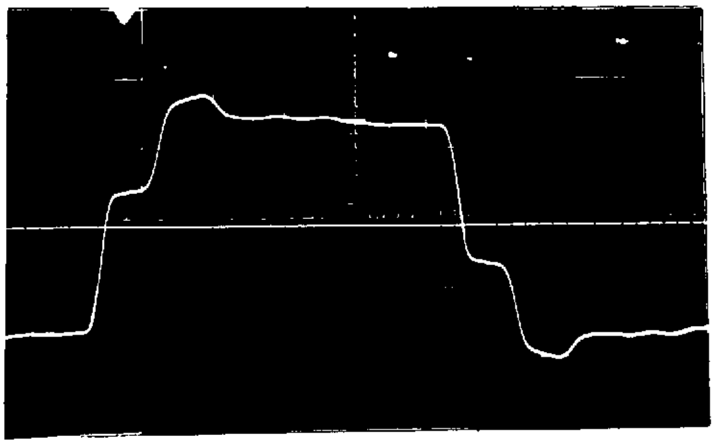
b. Receiver 1

Figure 9. Effect of One Stub With Much Greater Length Than Pulse Rise Time



a. WAVEFORMS AS SEEN BY RECEIVER 1

STUB = 5 TO 25 FEET



b. WAVEFORM AS SEEN BY RECEIVER 2

STUB = 25 FEET

Figure 10. Effect of Stub; Scale = 100 mv/div, 100 ns/div

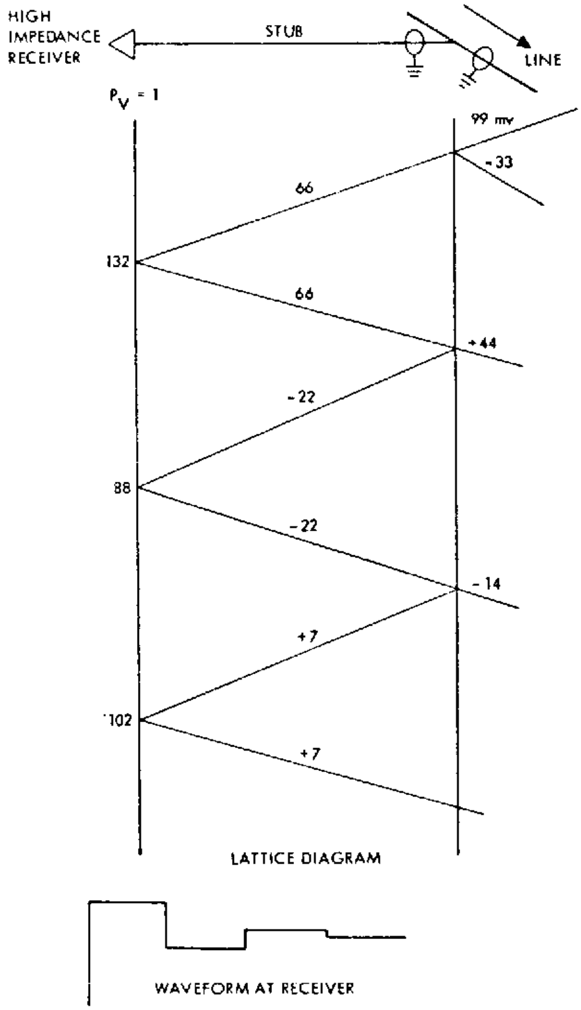


Figure 11. Lattice Diagram and Waveform at Receiver

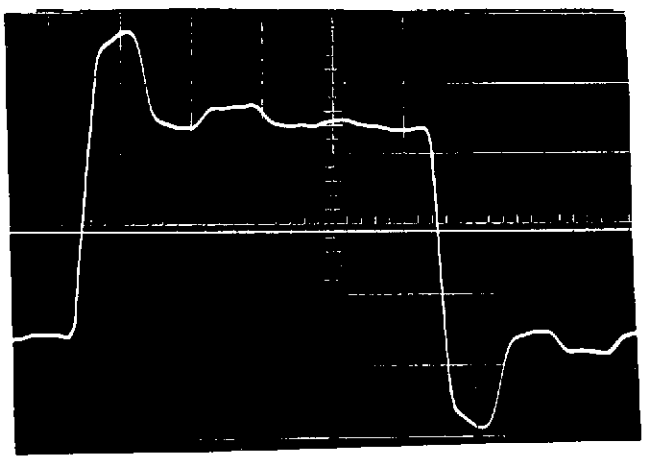


Figure 12. Receiver Voltage at End of 25-Foot Stub
Note agreement with expected waveform in Figure
Scale: 100 mv/div, 100 ns/div.

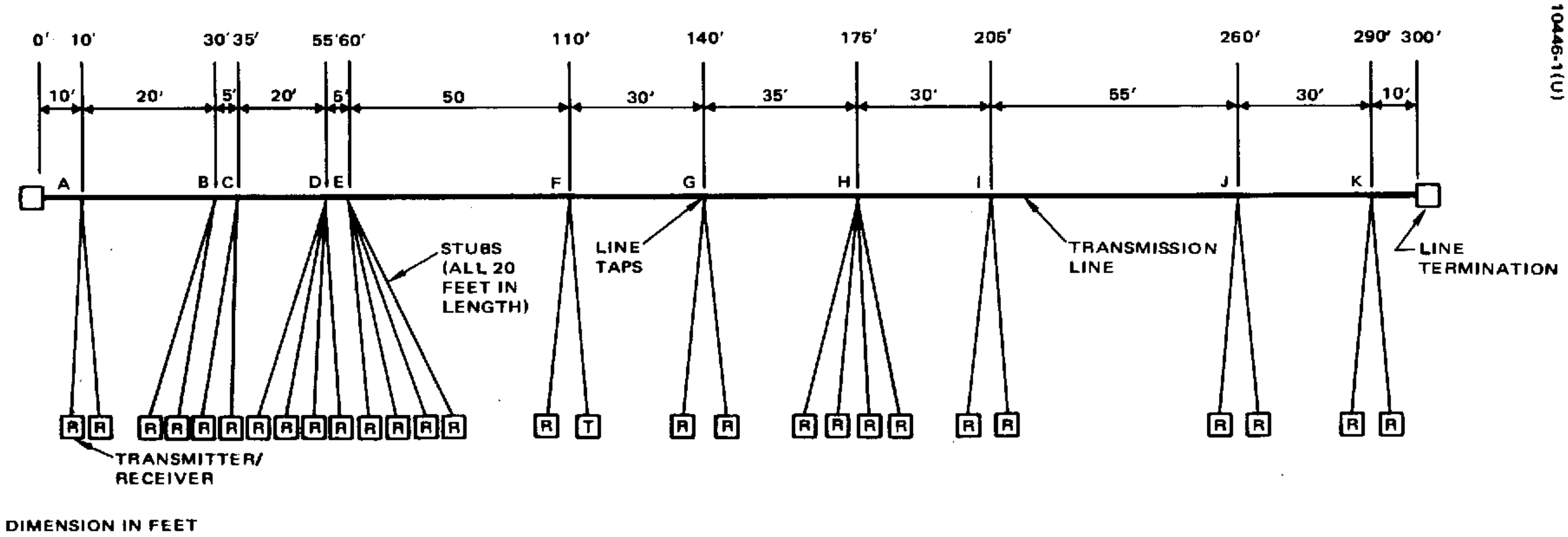
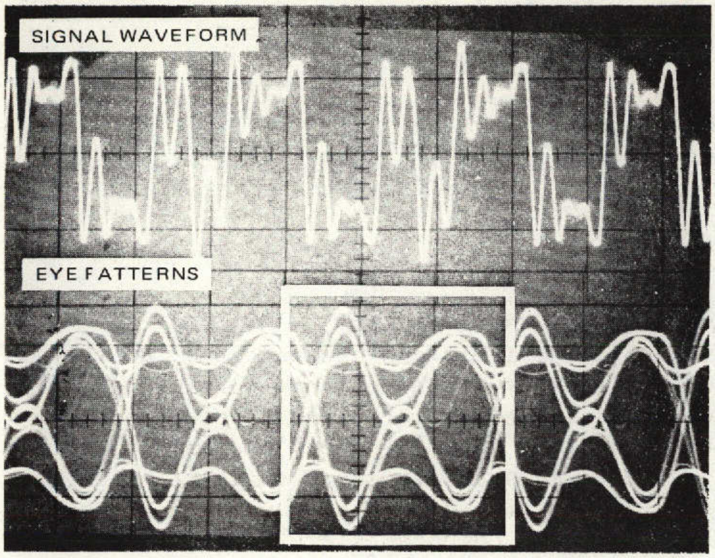


Figure 13. Laboratory Test Setup



10466-2(U)

VERTICAL: 2.0 VOLTS/DIVISION
HORIZONTAL, TOP: 1.0 µSEC/DIVISION
HORIZONTAL, BOTTOM: 0.2 µSEC/DIVISION

Figure 14. Waveform and Eye Patterns at Receiver G with Transmitter at F

The vertical dimension of the eye pattern opening indicates the minimum noise margin against noise when sampling the wave. The horizontal dimension indicates the amount of time jitter of transitions and the amount of peak distortion due to intersymbol interference.

The second echo contributors are the reflections of the signal on the main line between line tap terminations. At every tap junction a portion of the wavefront is reflected back to the source. When this reflected wave reaches another tap it is reflected again, thereby "trapping" the signal on the line between terminations, with a fraction of the energy being released at every reflection. Echoing effect is hereby produced as a result of the time delays of the released energy of the signal. Again, only the high frequency components are affected because the delayed signals are significantly shifted in phase from the original signals.

Figure 15 is a lattice diagram demonstrating the accumulation of phase shift as a result of the mismatched stubs. As can be seen from this diagram, the amount of phase shift involved is very much a function of the geometry of the system, including the relative location between the drivers and receivers. Even though this test was performed at relatively low frequency with rise time limited to 100 nanoseconds, the length of the line was far greater than the rise time (300 feet or 450 nanoseconds). In addition, the stub lengths were 20 feet or 30 nanoseconds. In other words, the line was approximately five times longer than the rise time, and the stub lengths were one-third the rise time. It was experimentally tested that minimum distortions will occur if the stub lengths are in the order of thirty times shorter than the rise time. Thus, for stub lengths less than six inches (approximately one nanosecond), the minimum rise time allowed is 30 nanoseconds.

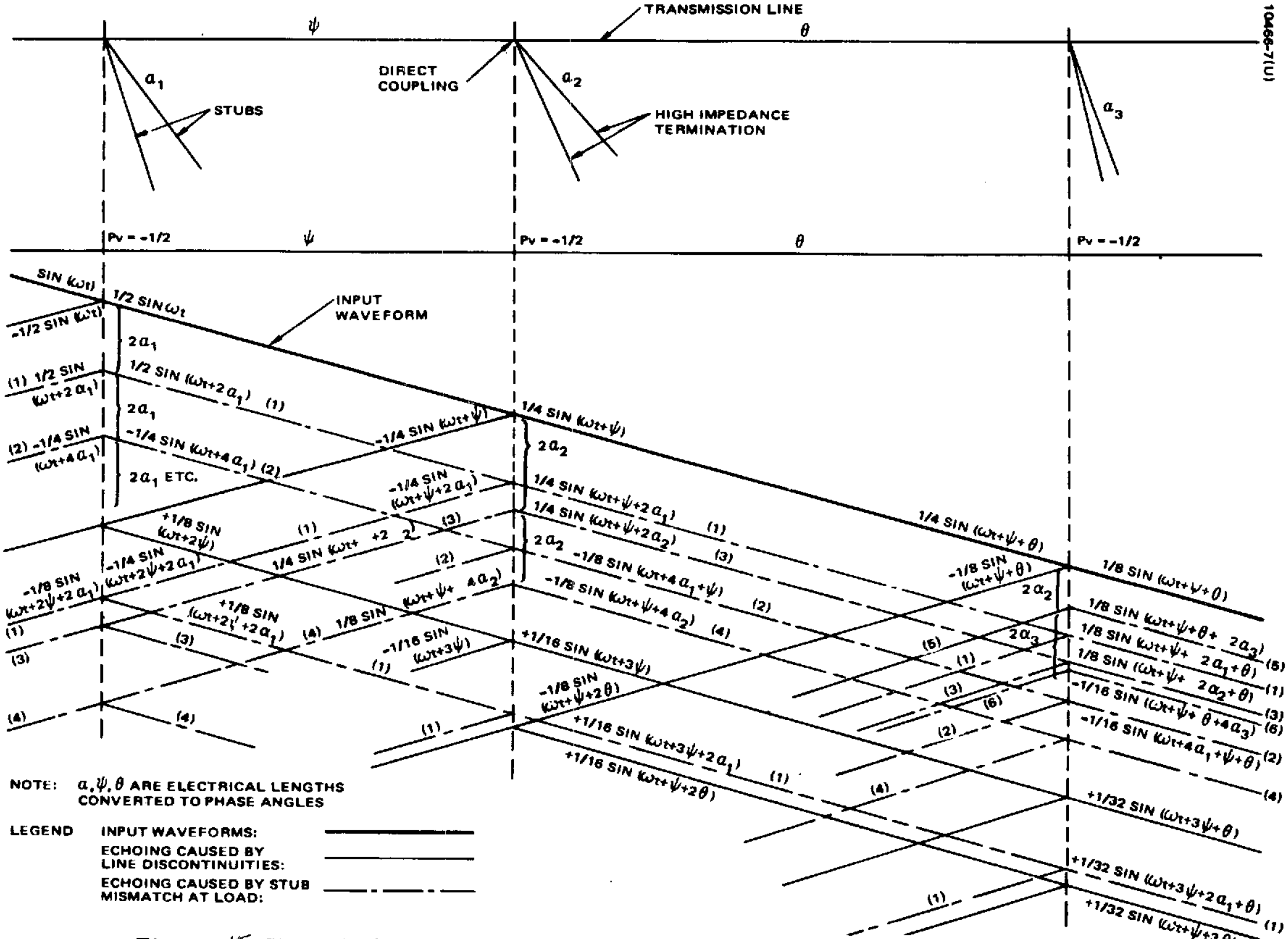


Figure 15 Simplified Lattice Diagram - Phase Distortion as a Result of Mismatched Stub Terminations is Clearly Demonstrated

4.0 TRANSMISSION LINE TERMINATION TECHNIQUE

To minimize the power dissipation through the transmission line terminations, it is desired to make the terminations as large as possible. However, to minimize reflections it is desired to match the terminations with the characteristic impedance of the line which is in the order of 100 ohms. With this dilemna in mind, a compromised solution to the value of the terminations can be reached provided the allowed distortion time and the line lengths are known. Figure 16 represents a lattice diagram of a data link terminated at both ends with termination resistors of value such that the coefficient of reflection is p. The voltage at any time is the sum of the initial voltage and the reflections up to that time of interest. When the distortion time and the line length are defined, only a particular number of reflections are allowed to reach a desired percentage of the final value of the line voltage. The following example will illustrate the procedure.

Example: Given a transmission line 4 feet in length. The data transmisssion is 10 MHz rate. The rise and fall time of the pulses is 15 nanoseconds. The condition is such that it is desired to reach 90% of the final voltage within 30 nanoseconds after the arrival of the pulse (i.e. 15 nanoseconds allowed for distortions). (See Figure 17). It is desired to compute the maximum allowable value for the termination resistors.

Solution: The line length is 4 feet or 6 nanoseconds. Within 15 nanoseconds two reflections can be allowed. Thus the following equation can be written:

$$V_I \,(1 + p + p^2) = .90\, V_F; \text{ where } V_I = \text{initial voltage}$$

$$V_V = \text{final voltage}$$

Assume current drive,

$$V_F = I_o \frac{Z_L}{2} = \frac{I_O Z_O}{2} \; \frac{(1 + p)}{1 - p} \quad ; \; V_I = \frac{I_o Z_L}{2}$$

Thus,

$$\frac{I_O Z_O}{2} \,(1 + p + p^2) = (.90) \left(\frac{I_O Z_O}{2}\right) \frac{(1 + p)}{1 - p}$$

solving for p

$$(1 - p)\,(1 + p + p^2) = .9\,(1 + p\_$$

$$(1 - p^3) = .9\,(1 + p)$$

since $p^3 << 1$, $1 - p^3 \cong 1$ and

$$\frac{1}{.9} - 1 = p = \frac{1 - .9}{.9} = \frac{.1}{.9} = .11$$

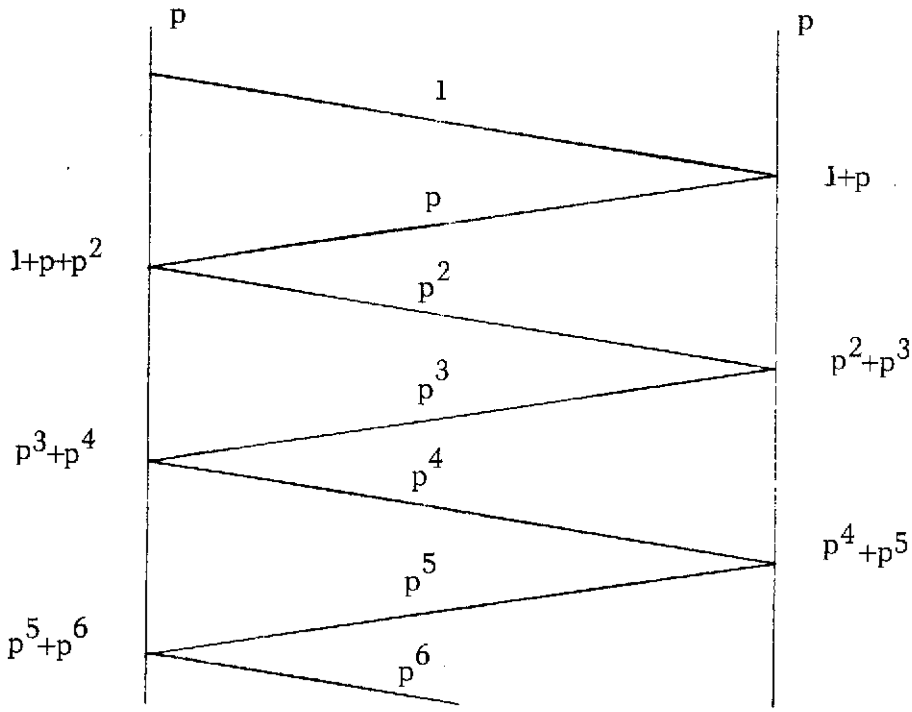


Fig. 16 Data Link Lattice Diagram

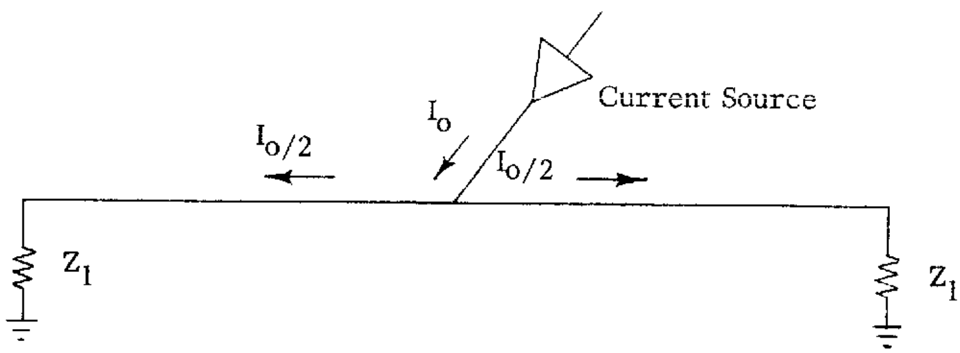


Fig. 17 Sample Problem Setup

$$Z_L = Z_0 \frac{(1 + p)}{1 - p} = Z_0 \frac{(1 + .11)}{1 - .11} = 1.25Z_0$$

Thus the termination resistor can be one and a quarter times greater than the characteristic impedance of the line.

5.0 PHYSICAL DESIGN ASPECTS

In an attempt to define the maximum wire length on the data buses, several assumptions must be made concerning the number of modules and their size. With the present information it will be assumed that each processor module will be approximately 12 cubic inches in volume and will measure 3 X 4 X 1 inches. This volume will allow the interconnections of between 27 and 45 high density logic arrays. Exact component density will be a function of the number of input/output connections per array. Each memory module is assumed to be 6 X 4 X 1 inch. Allowing four inches in width for unit wiring, common circuits and power supplies and assuming 16 processor and memory modules, we arrive at a unit size of 16 X 17 X 8 inches. With the further assumption that the data bus lines must input to each module, we arrive at a maximum wire length of 42 inches.

Because of the small size of both the processor and memory modules, limitations do exist on the number of bus lines which may terminate in each module and on how they may terminate. If a bit orientated data bus scheme is adopted, "tapped" or "daisy chain" connections must be eliminated because this system will double the number of input/output lines associated with each module. While this interconnection method minimizes the discontinuities in each line, the wire bulk in the connector area of the modules will make this system unworkable. The more acceptable interconnection method is the tree fanout system with each branch of the tree limited to less than one inch. This short stub length should provide no significant reflection on the line.

The question of how many bus lines must be used to achieve the 200 megabit per second goal must now be answered. Since power and weight are to be minimized, it does not appear practical to use high speed TTL or ECL. The significant power consumption of these devices over CMOS or the series 7400 TTL will cause a significant weight gain in the final hardware. Figure 1 represents the power dissipation per gate for various hardware families. From the graph a practical transmission rate from a power point of view is 25 megabits per second per data line. This speed then establishes a lower bound of 8 data bus lines. If a bit orientated data bus scheme is adopted, the upper bound for transmission lines is then 32 lines. Operating at a data rate of 6.7 megabits per second. With these bounds, we are limited to basically three logic systems, 1) MOS - either P-MOS ion implanted or C-MOS, 2) Series 7400 T^2L, or 3) low power Schottky T^2L.

Detailed weight tradeoffs will have to be performed to clearly establish the most efficient logic system from a hardware point of view. At speeds up to 10 megabits per second, a large question arises as to whether low power TTL or MOS is the most efficient. The ability to design and produce large scale MOS arrays (those with active device counts greater than 500) makes the MOS much more attractive than TTL. It does not seem unreasonable to have the higher power dissipated by the MOS device completely overshadowed by the significant reduction in interconnection it possesses. At the higher speeds, those greater than 10 megabits, device complexity and interconnection minimization will be the principal items to be traded-off.

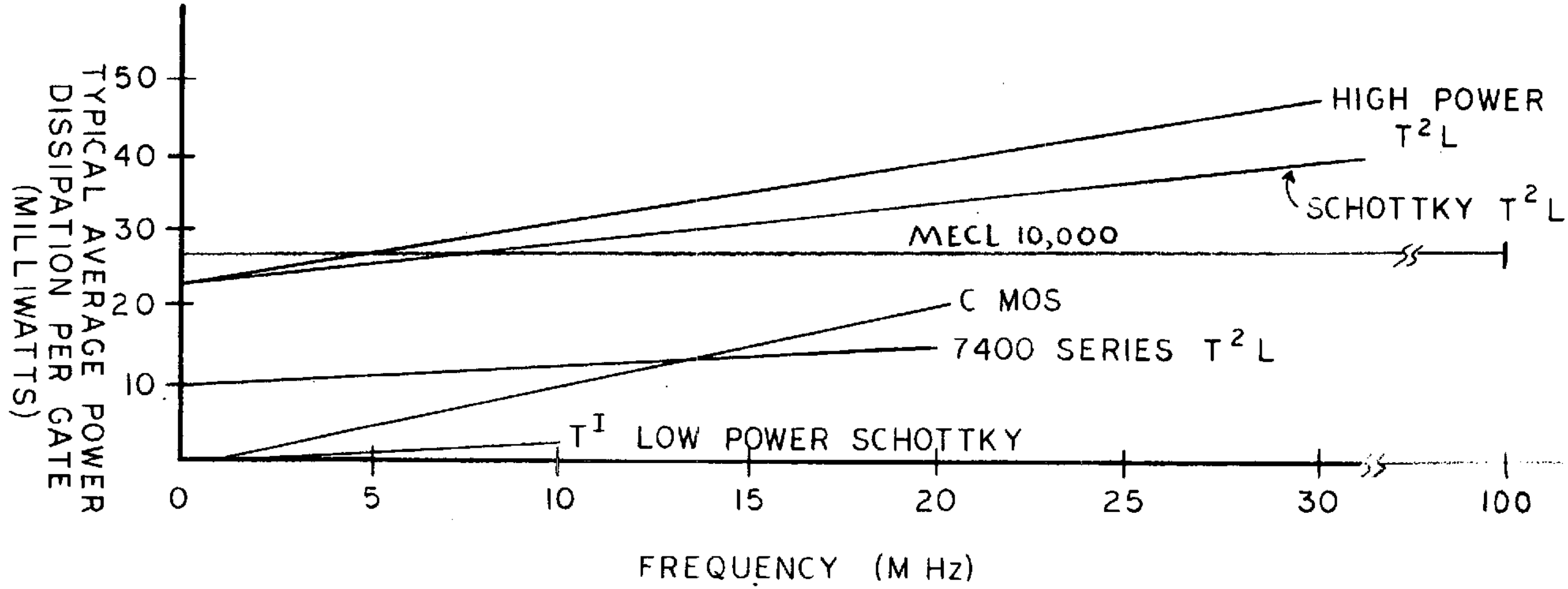


FIG. 1 POWER DISSIPATION VS FREQUENCY OF OPERATION

With the above bound on transmission speeds, it is now necessary to discuss the merits of the various wiring systems. Those which present promise for the data transfer lines are 1) wire over ground, 2) twisted pair, both shielded and unshielded, 3) ribbon cable, 4) contour cable with stripline or microstripline transmission lines, and 5) coaxial cable.

The wire over ground possesses the unique property of being the fastest (shortest propagation time) of the commonly employed transmission lines. Several practical problems exist however, when attempting to employ this concept in high density microminiature hardward. Proximity of each line to one another can create significant crosstalk between adjacent lines. Physical support and maintaining proper separation to ground presents a problem during unit assembly and test. The system may also be susceptible to failure during fibration and acoustical noise testing. These problems can be overcome by supporting the wire in a low dielectric foam, but the problem of crosstalk remains. With high wire density the bulk of wire may make assembly difficult. For these reasons the wire over ground is not considered the most practical system for the data bus lines.

Twisted pair and shielded twisted pairs have been used successfully for handling digital signals up to 15 megabits. Beyond this frequency, line losses are sufficiently large to prevent transmission of large bandwidths over long distances. The addition of a shield to a twisted pair significantly reduces the problem of signal crosstalk to other lines. Assuming that the center-to-center spacing of one inch is maintained between modules, the actual length of twisted wire would be reduced to a small fraction of an inch. In a practical sense in this hardware, the twisted pair degenerates to a wire over ground system. For this reason, twisted pairs do not appear to be a practical solution to the wiring problem.

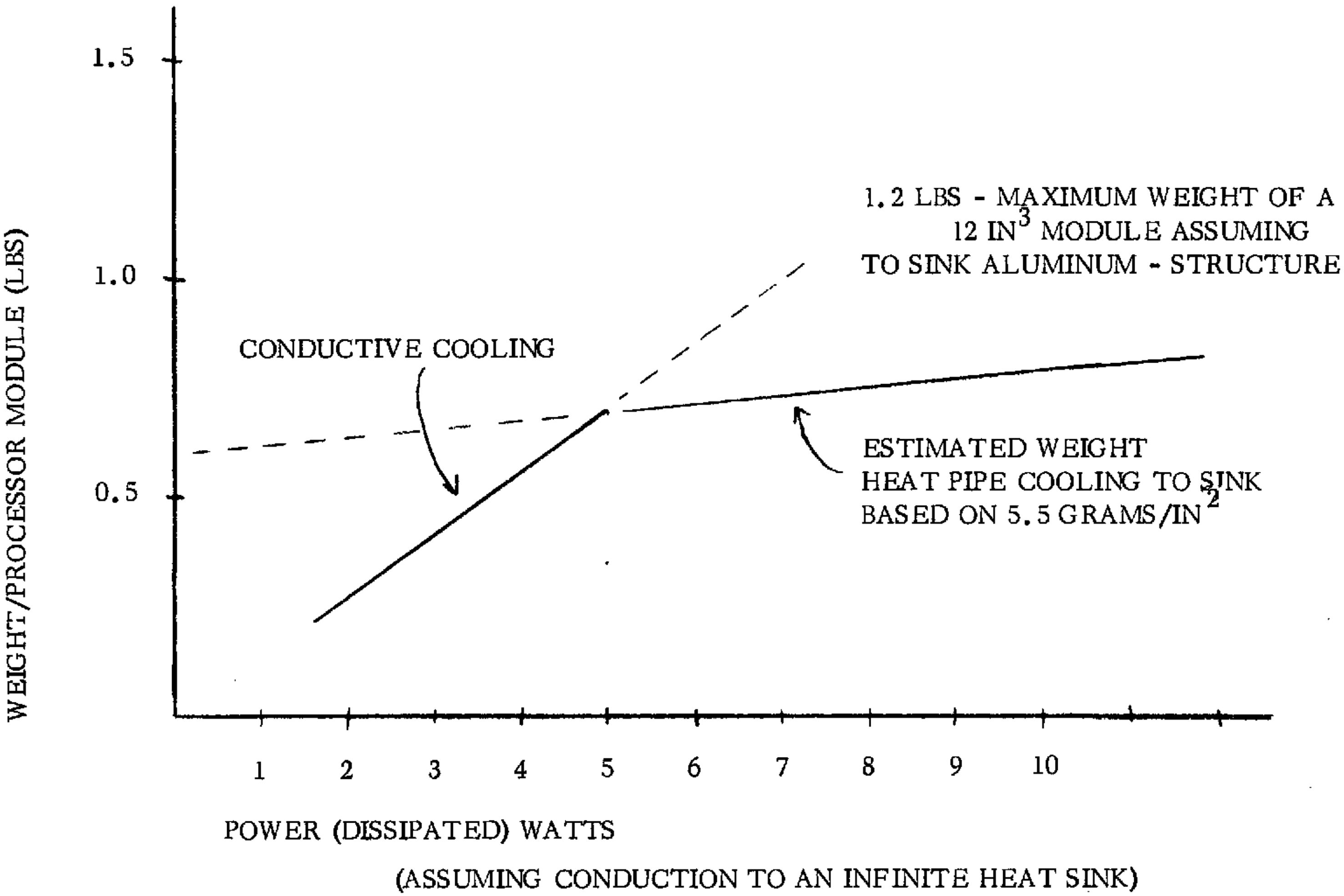
Ribbon cable or strip cable is possible the first practical system for data transmission in this type of hardware. It offers the advantage of being able to be designed as a transmission line with a large variety of impedances, crosstalk between adjacent lines can be minimized with proper design, and attenuation can be eliminated as a problem by the proper choice of ground and signal conductor sizes. Cables of FEP and FEP-Polyamide have successfully been employed in numerous spacecraft systems since the origination of United States space programs. Ribbon cable does, however, have several practical drawbacks. Because of its design it may be difficault to handle and install. With close spacing, termination could be a problem and stub length could become excessive. While the problems of installation and termination make the use of ribbon cable difficult, it is not without merit and should not be discounted when considering types of transmission systems.

There does exist however a system which possesses all of the desireable properties of the ribbon cable with few, if any, of its drawbacks. The use of an etched contour cable employing either a stripline or microstripline technique appears to be best suited for data transfer in this system. Either system will exhibit propagation delays of approximately 2 nanoseconds per foot with crosstalk between adjacent signal lines of 2% or less. By proper design, characteristic impedances of 50 to 100 ohms may be designed with DC resistances of less than $\frac{1}{2}$ ohm (assuming a 42 inch run). The use of this wiring technique allows the circuit a wide latitude of drive networks with a minimum hardware implication. One wire or two wire transmission systems can be employed with equal ease and without major problems of interconnection to the modules. Stub length can be reduced to fractions of an inch and thereby reduce line reflections.

Because of the ease of installation, the ease of controlling characteristic impedance of the signal conductors, the elimination of crosstalk problems, the low propagation delays and low DC resistance, the stripline contour cable appears to be the most efficient wiring system for the data bus transmission lines.

Miniature coaxial cable could be used as a transmission system since it possesses all the desirable characteristics of speed and isolation needed. It does have several drawbacks which eliminate it from consideration in this system. The physical size of the sub-miniature OSSM connector will prevent more than 8 data lines being terminated in each module without designing new ganged connectors. Second, the physical separation of the modules (one inch) may make cable installation impossible without unduly increasing the total wire length. For these reasons miniature coax is not recommended as the most desirable data bus transmission system at this time.

In the thermal design of a spacecraft system, electronic component cooling is accomplished by conduction or radiation of power to a low temperature sink. Conduction cooling has proven to be the most efficient method of removing small amounts of power from point sources (electronic components). It is not without limitation, however. With an individual module, as power increases material thicknesses must increase proportionally to maintain a fixed T between a component and its thermal sink. If the thermal control structure thickness for a given module is 0.06 inches and the power dissipated within the module is 1.3 watts, doubling the power would require twice the thermal structure thickness to maintain a given set of boundry conditions. With these conditions in mind, a practical power limit appears to be $4\frac{1}{2}$ to 5 watts in each processor module employing conduction cooling. Above these levels, thermal control requires more than half of the volume allocated to the module. (Reference figure 2). At this point more sophisticated thermal control systems must be employed. The use of a small heat pipe on each of the printed circuit boards within the module would increase the weight of the module to approximately 0.7 pounds but would allow the components to operate near the same temperature as the thermal sink. With this arrangement, power levels as high as 30 watts could be dissipated within the module and still maintain semi-conductor junction temperatures below 150°F with a sink temperature of 70°F.



PROCESSOR WEIGHT VS POWER DISSIPATION

FIG. 2

APPENDIX A

A UNIFIED METHOD FOR ANALYZING MISSION PROFILE RELIABILITY FOR STANDBY AND MULTIPLE MODULAR REDUNDANCY COMPUTING SYSTEMS WHICH ALLOWS FOR DEGRADED PERFORMANCE.

This appendix is the manuscript of a paper written by J. J. Bricker which describes the second reliability model developed for the ARMMS program. It is being submitted for publication to the IEEE Transactions on Reliability Theory.

The purpose of the model is to allow the reliability of various ARMMS configurations to be predicted allowing for phase changes at deterministic time points when the number of modules per class may be changed. Each module of a given class has both an active and a passive failure rate where it is assumed that negative exponential distributions of active and dormant failures prevail throughout and the given class may operate either in NMR ($n+1$ out of $2n+1 = N$) or as a standby sparing system.

The analysis proceeds by generalizing the notions of standby and NMR redundancy, which for $N = 3$ is TMR (Triple Modular Redundancy), into a concept called hybrid-degraded redundancy. The probabilistic evaluation of this unified redundancy concept is then developed, via Laplace Transforms, to yield, for a given modular class, the joint distribution of success and the number of non-failed modules from that class, at special times. With this information, a Markov chain analysis gives the reliability of an entire sequence of phases or mission profile.

The model has been programmed and various sample mission phase profiles are being explored.

A UNIFIED METHOD FOR ANALYZING MISSION PROFILE RELIABILITY FOR STANDBY AND MULTIPLE MODULAR REDUNDANT COMPUTING SYSTEMS WHICH ALLOWS FOR DEGRADED PERFORMANCE

J. L. Bricker*
Hughes Aircraft Company
Fullerton, California

I. Introduction: A. The Main Purposes of the Analysis

A modular computer concept which is responsive to both large computing capacities and high reliability objectives is being explored by the Marshall Space Flight Center of NASA. This system, designated ARMMS (Automatically Reconfigurable Modular Multiprocessor System) is to allow its constituent modules to be utilized as parallel, simplex computers for high throughput or the system can be reconfigured to employ some modules in the Triple Modular Redundant (TMR) mode for maximum reliability in performing critical computations. The reliability of such a computer is dependent on the sequence of mission phases and the complement of modules which must be active in each phase. The computer under study is contemplated for use on deep space scientific missions or in orbital space stations. On a mission to the outer planets a high level of reliability must be achieved without human intervention in the form of repair capability or logistic support, hence the need for a general detailed modelling effort.

A complete analysis of this problem has been developed under the classical assumptions of independence of module failures and negative exponential failure laws for each module, either in the active or standby states. It is also assumed that intra or inter

*This work was sponsored by the Marshall Space Flight Center of the National Aeronautics and Space Administration through Contract NAS8-27926 with the Hughes Aircraft Company, Fullerton, California.

modular class switching, failure detection and location and switch-off and switch-on transients (the so-called factor of coverage) behave perfectly and that the voter reliability is equal to one.

Thus, for example, although the parameter of coverage, c, (See [2], P. 200) is known to play an important role in the overall analysis, as far as active-standby redundancy is concerned, it was considered premature to attempt to incorporate the known formulas involving c into the modelling effort, without properly considering the individual effects of failure detection, location, switching and power-on or power off transients, as they pertain to the ARMMS design.

The analysis unifies the existing treatments of TMR and standby redundancy in the literature (See [1], [2], [3], [6], [7], [8]) and shows that these two types of redundant organization have basically the same reliability analysis, if one extends the hybrid concept of redundancy to include degraded modes of operation.

This task is actually required in order to properly perform the overall mission profile reliability calculation and is not included for reasons of mathematical elegance. In addition, it assists in numerical and programming efforts since one set of equations is derived and not two apparently unrelated ones.(1)

(1)[Compare, e.g., Eq 8, P. 380 of [7] with the equation for cR_s^q (T, λ, μ) on P. 1308 of [6] or with the equation for $R_{TMR/S}$ on P. 1307 of [6]].

B. Some Basic Terminology and Symbols and Pertinent Literature Review

In [1] and [7], Mathur has provided some convenient terminology and symbols which we shall use.

When discussing differing module classes, the symbols below may be indexed by a subscript.

λ	Failure rate of an active module = Power on failure rate.
μ	Power off failure rate = Failure rate of a dormant module (we assume $\mu > 0$ but the case $\mu = 0$ can be treated by taking limits using L'Hospital's Rule).
$K = \lambda/\mu$	$1 \leq K < \infty$ is of most interest.
S	Total number of standby spare units, $S \geq 0$.
$N = 2n+1$	Total number of active redundant units = 3 if the system is TMR.

n = Degree of active redundancy = 1 in TMR system.

$C = N+S$	Total number of units in a module class.
T	Mission Time, (≥ 0)
t, τ	Dummy variables for time $0 \leq t, \tau \leq T$.
Simplex System	A non-redundant module (in a system consisting of several module classes, a simplex would be a set or chain of modules with exactly one from each class).
TMR System	Triple-modularly redundant system (N=3) - Two-out-of-three majority logic.

NMR System	N-tuple-modularly redundant system - one in which majority logic or n+1 out of 2n+1=N is used.
Hybrid (N, S) System	A hybrid redundant system having a total of N+S units in which N units are active and majority logic is used and S spares are standby spares.
H(N, S)	An abbreviation for Hybrid (N, S).
A(L, S)	The classic active-passive standby system in which L modules are always active and the system fails after all possible spares are exhausted and only L-1 active modules are left (no majority logic employed here, i.e., no voting).
H(3, S)	A TMR system with S spares.
R(N, S) [T]	The reliability of an H(N, S) system of duration T.

In [1], Mathur developed the analysis for:

$$R(3,S)\,[T] = R_{TMS/S}$$

$$= 3R^2\left\{\prod_{i=0}^{S-1}\frac{(3K+S-i)}{(K+S-i)} - \frac{2RK^2}{S!}\left[\prod_{i=0}^{S-1}(3K+S-i)\right]\sum_{i=0}^{S}\binom{S}{i}\frac{(-R_s)^{S-i}}{(K+S-i)\cdot(3K+S-i)}\right\} \qquad \text{(Eq 1)}$$

where

$$R = e^{-\lambda T}$$

$$R_s = e^{-\mu T}$$

Later, in [7], he generalized the result to R(N, S) [T] to obtain

$$R(N,S)\,[T] = R^N R_s^S \Bigg[1 + \sum_{j=0}^{S-2} \binom{NK+S}{j+1}\left(\frac{1}{R_s} - 1\right)^{j+1} + \sum_{i=0}^{n} \binom{N}{i}\binom{NK+S}{S}$$

$$\sum_{\ell=0}^{i} \frac{\binom{i}{\ell}(-1)^{i-\ell}}{\binom{K\ell+S}{S}} \left\{ \left(\frac{1}{R_s^S R^\ell} - 1 \right) - \sum_{j=0}^{S-2} \binom{K\ell+S}{j+1}\left(\frac{1}{R_s} - 1\right)^{j+1} \right\} \Bigg] \quad \text{(Eq 2)}$$

The problem of A(L, S) was initially solved by Kletsky, [9], rediscovered by Bouricius, etal, [2], and there generalized by the latter to include the factor of coverage, c, where:

c = Conditional probability that after a failure in the system (either in any of the active modules or spares), the system is able to recover and continue information processing with no permanent loss of essential information.

= Prob (that fault detection, location and appropriate switching functions are properly performed in the advent of a failure, for the A(L, S) type of redundancy).

= Prob (appropriate switching – out of the bad module and in of the good module in H(N, S) systems), since the other basic functions are presumably hardware functions of properly designed voter units.

Bouricius, [2], obtains for $R_A(L, S)$ = Reliability of A(L, S), the expression:

$$R_A(L,S) = e^{-L\lambda T} \sum_{k=0}^{S} \binom{k-1+LK}{k} c^k \left(1 - e^{-\mu T}\right)^k \quad \text{(Eq 3)}$$

We now define a hybrid-degraded system H(N, S, D), where N need not be an odd integer and $0 \le D \le N$, as follows: Initially there are N+S modules in the system in which N are active and S are dormant spares. The system runs until only N active units are still functioning, at which point it continues to operate until only $D \geq 0$ units operate. When the number of active units falls to D-1 the system is considered to have failed but one can still consider the system to be running until all modules are down. (Note: If $D=0$, the system never fails). Since for large intervals of time during the mission, modules may be turned off to conserve power and, presumably, to enhance the effect of redundancy, since power-off failure rates are usually presumed to be lower than power-on failure rates, we must allow for $D=0$, as a possibility. For example, when the spacecraft is traveling between planets and there are no important scientific experiments to consider, the computer modules of several module classes may be turned off. Observe that if one assumes perfect coverage or perfect voter and switching reliability, then both H(N, S) and A(L, S) are special cases of H(N, S, D) as far as reliability is concerned. In fact, for $N=2n+1$,

$$H(N, S) = H(N, S, n+1)$$

while, for L arbitrary,

$$A(L, S) = H(L, S, L)$$

In the classical model of n+1/2n+1 voting, it should be clarified that once all the spares are exhausted and the system subsequently drops from 2n+1 to 2n operative units, one maintains the criterion that n+1 modules must agree. Additional failures, which we always assume to occur only one at a time, won't hurt the system performance therefore, until the system has only n+1 remaining modules. The next module failure will <u>then</u> be a system failure.

The term D is introduced to allow for module class mode degradation. Suppose, for example, that in a TMR/S system, i.e., H(3, S), one operates as follows:

As long as two voters agree (see Taylor, [10]) the voter assumes that the third has failed and thus will switch in a previously unpowered spare, if one exists. If no spare is available, the system can still operate as long as the two remaining units agree. However, when they disagree, the voter cannot tell which unit is in error and for that reason the voter is useless from this time on. Since one does not wish a system to fail if one good unit remains, software error detection may be introduced at this point to determine the faulty unit. Through the proper use of software error detecting schemes, (assuming perfectly reliable software, i.e.) the final unit could then be utilized until it fails.

Denoting by H*(3,S) the above hybrid-hardware-software-redundant procedure, then, clearly, we again have a special case of the scheme H(N,S,D) where,

$$H^*(3, S) = H(3, S, 1)$$

Thus, a complete reliability evaluation of the hybrid-redundant scheme H(N,S,D) will encompass all the previous seemingly unrelated type redundancy schemes. In this sense, in the absence of an exact design for performing coverage and replicated voting and switching, one obtains a unifying principle for studying some of the more popular redundancy configurations.

There is yet another type of hybrid-redundancy called Hybrid-Simplex Redundancy, [8], which for N=3 is then TMR/Simplex Redundancy, in which one operates as H(3, S) until two modules are left, and then one of the two remaining units are discarded, so that from that time on the system is operated in a simplex mode. This might be done in the

C 5

absence of a good software error detecting package so that, since the voter could not distinguish a failure among the two remaining modules, one may as well be discarded to simplify the executive decision logic. By the introduction of yet another two parameters D^1, D^2, one could consider TMR/Simplex as an $H(N, S, D^1, D^2, D)$ system where the latter notation means that one runs an $H(N, S)$ system until D^1 active units are left, at which point D^1-D^2 of these are discarded and the final system consisting of D^2 active units are run until the system has only $D-1$ left. All the above considered schemes are special cases of this 5-parameter family of schemes whose reliability function is as easy to derive as that of $H(N, S, D)$. Thus, for $N=2n+1$ one has,

$$H(N, S) = H(N, S, n+1, n+1, n+1)$$

$$A(L, S) = H(L, S, L, L, L) \text{ or in general } H(N, S, D) = H(N, S, N, N, N)$$

which is equivalent to an $H(N, S, D, D, D)$ scheme

$$\text{TMR/SIMPLEX} = H(3, S, 2, 1, 1)$$

Since no use of TMR/Simplex redundancy is planned in the ARMMS design, it is mentioned only for completeness and to point out the extension and generality obtainable from the present analysis.

II. Derivation of $R(N, S, D)[T]$ and $P_i(N, S, D)[T]$

We now proceed to obtain some preliminary results before precisely restating the MPRP (mission profile reliability problem). We shall require for the MPRP not only $R(N, S, D)[T]$ = Prob of success of the $H(N, S, D)$ scheme at time T, but also, Prob (That the number of non-failed modules = i when the module class is operated as an $H(N, S, D)$ scheme in the time interval $[0, T]$), which is denoted by $P_i(N, S, D)[T]$, where $D \leq N$, for $0 \leq i \leq N+1$. If $i \geq D$, $P_i(N, S, D)[T]$ represents the Prob (That there exists a degraded mode of success of level i at time T).

Problem Statement: Assume that there are N+S identical modules of which N are simultaneously active and S are in standby at time 0, all being functional. Let each of the active modules fail according to the negative exponential failure law,

$$P(\tau_1 < t) = F_1(t) = 1 - e^{-\lambda t}, \quad \text{for} \quad \lambda > 0$$

while each of the standby modules fails with the negative exponential failure law

$$P(\tau_2 < t) = F_2(t) = 1 - e^{-\mu t}, \quad \text{for} \quad \mu > 0$$

Here τ is the random variable denoting time until failure occurs when the module stays in the appropriate state (either active or standby). Assume further that all modules fail independently of one another, and that the system operates by switching spares into the bank of N operating modules as long as spares are available, but that once the number of non-failed modules reaches the number N the system uses all available modules until D are left. The system will then be considered in a degraded mode if $D < N$ and will be considered in the failed mode when the number of non-failed modules reaches D-1. We determine,

$$P_i(N, S, D)\,[T] = \text{Prob (At time T the number of non-failed modules} = i)$$

Following, Bricker [11], let us define the random variables:

$$\left\{ X_i \right\}_{i=1}^{N+S},$$

where X_i = Time from the instant of the (i-1) st failure, occurring in the system H(N, S, D), until the instant of the ith failure,

and let

$\#_t$ = Number of system failures (spares or operating) in the time interval [0, t].

$$T_i = \sum_{j=1}^{i} X_j = \text{Time of the i-th system failure}$$

Then the event $[T_i > t]$ is equivalent to the event $[\#_t \leq i-1]$

Since,

$$[\#_t \leq i] - [\#_t \leq i-1] = [\#_t = i]$$

$$\text{Prob}\ ([\#_t \leq i]) - \text{Prob}\ ([\#_t \leq i-1]) = \text{Prob}\ ([\#_t = i])$$

$$\text{Prob}\ ([T_{i+1} > t]) - \text{Prob}\ ([T_i > t]) = \text{Prob}\ ([\#_t = i])$$

or

$$(1 - F^{i+1}(t)) - (1 - F^{i}(t)) = P_{N+S-i}(N, S, D)\ [t] = F^{i}(t) - F^{i+1}(t), \text{ for } 0 \leq i \leq N+S \qquad \text{(Eq 4)}$$

where $F^{(i)}(t) = \text{Prob}\ (T_i \leq t)$ and $F^{(0)}(t) < 1$, by definition.

Let $f^{(i)}(t)$ = Probability density function of the random variable

$$T_i = \sum_{j=1}^{i} X_j \qquad \text{for } 1 \leq i \leq N+S$$

Now X_i is a negative exponentially distributed random variable with failure rate λ_i where

$$\lambda_i = N\lambda + (S-(i-1))\mu, \qquad \text{for } 1 \leq i \leq S+1, \qquad \text{(Eq 5a)}$$

while for $S+1 \leq i \leq N+S$ we have a system consisting entirely of active modules so that

$$\lambda_i = (N+S+1-i)\lambda, \qquad \text{for } S+1 \leq i \leq N+S \qquad \text{(Eq 5b)}$$

Thus, e.g. $\lambda_{N+S} = \lambda$.

Since $\mu,\ \lambda > 0$ we see that: $\lambda_1 > \lambda_2 > \lambda_3 > \dots > \lambda_{S+N} = \lambda > 0$ so that all the system (module class) failure rates are distinct.

Let

$$\mathcal{L}(f_i(t)) = \int_0^\infty e^{-\sigma t}\,(\lambda_i e^{-\lambda_i t})\,dt = \frac{\lambda_i}{\sigma + \lambda_i}$$

be the Laplace Transform of $f_i(t)$, for $\sigma > 0$, where $f_i(t)$ is the pdf of the random variable X_i.

$$\therefore\ \mathcal{L}(f^{(i)}(t)) = \prod_{j=1}^{i} \mathcal{L}(f_j(t)) \quad \text{since } T_i = \sum_{j=1}^{i} X_j,$$

by the convolution theorem for Laplace Transforms, and one finds,

$$\mathcal{L}(F^{(i)}(t)) = \frac{1}{\sigma} \prod_{j=1}^{i} \frac{\lambda_j}{\sigma+\lambda_j} = \frac{1}{\sigma} \sum_{j=1}^{i} \frac{A_j^i\,\lambda_j}{\sigma+\lambda_j} \qquad \text{(Eq.5c)}$$

using partial fraction decomposition.

Therefore, by a standard argument one obtains:

$$A_j^i = \prod_{\substack{k \neq j \\ 1 \leq k \leq i}} \frac{\lambda_k}{(\lambda_k - \lambda_j)} \tag{Eq 5d}$$

Then for $1 \leq i \leq N+S$,

$$f^{(i)}(t) = \mathcal{L}^{-1}\left(\sum_{j=1}^{i}\right)\frac{A_j^i \lambda_j}{\sigma + \lambda_j} = \sum_{j=1}^{i} A_j^i \lambda_j \; e^{-\lambda_j t} \tag{Eq 5e}$$

$$\therefore \; F^{(i)}(t) = \int_o^t f^{(i)}(x)\, dx = \sum_{j=1}^{i} A_j^i \left(1 - e^{-\lambda_j t}\right) \tag{Eq 5f}$$

Letting $t \rightarrow \infty$, we see that:

$$\sum_{j=1}^{i} A_j^i = 1$$

Hence,

$$P_i(N, S, D)\,[t] = F^{N+S-i}(t) - F^{N+S-i+1}(t) = \sum_{j=1}^{N+S-i+1} A_j^{N+S-i+1} \; e^{-\lambda_j t}$$

$$- \sum_{j=1}^{N+S-i} A_j^{N+S-i} \; e^{-\lambda_j t} \tag{Eq 5g}$$

$$P_i(N,S,D)\,[t] = \sum_{j=1}^{N+S-i} \left(A_j^{N+S-i+1} - A_j^{N+S-i}\right) e^{-\lambda_j t} + A_{N+S-i+1}^{N+S-i+1}\, e^{-\lambda_{N+S-i+1} t} \qquad \text{(Eq 5h)}$$

Clearly,

$$R(N,S,D)\,[t] = 1 - F^{(N+S-D+1)}(t) = \sum_{j=1}^{N+S-D+1} A_j^{N+S-D+1}\, e^{-\lambda_j t} \qquad \text{(Eq 5i)}$$

We will not enter into numerical analysis questions at this time, but it should be pointed out that the $\left\{A_j^i\right\}$ are sensitive to small values of μ, since the nature of the solution changes as $\mu \to 0$.

III. The Markov Analysis of MPRP

A. The General Setting

Consider a mission profile defined by a sequence of consecutive time intervals $I_1\ [0,\ T_1]$, $I_2\ [T_1,\ T_2]$, . . . $I_J\ [T_{J-1},\ T_J]$ into which the mission is broken, depending upon the critical functions which the spacecraft must perform. Phase 1, occurring in $I_1\ [0,\ T_1]$, could be the launch of the spacecraft, Phase 2 might consist of initial and midcourse guidance, etc. and since the trajectory of the spacecraft may be considered to be predetermined, one may assume, at least for the purposes of the present study, that the times T_1, T_2, . . . T_j, . . . T_J are deterministic (non-random). Phase j occurring during the interval $I_j\ [T_{j-1},\ T_j]$ could be a planetary flyby during which various scientific sensors must function properly, etc. It is now a design analysis problem to "size up" the mission profile by specifying what kind and type of computer activity is required during the Phase I_j. This must be done on a module by module basis so that one, in effect, will be specifying a "shopping list" of computer module requirements, in terms of the number of modules of each type required, type of redundancy to be employed for each module/class/phase and the duration of on-time that each module must be utilized. If during a given Phase I_j there will be several changes of the computer structure, then, for the determination of mission reliability, it may be necessary to further subdivide each mission phase so that the amount of computer hardware needed is fixed for that subphase. Assume that this has been done so that we shall now refer to a mission "phase" as a time interval $I_j\ [T_{j-1}, T_j]$ during which the number of modules of a class required is constant, the type of redundancy is specified (either TMR, or active-passive, etc) and the level of tolerable mode degradation, D, is determined, beyond which that phase will be deemed a failure. One then wishes to determine the reliability of the complete mission profile, viz:

$$R(\text{Mission }(J)) = R\ (I_1, I_2, \ldots I_J) = \text{Prob (Phase } I_j \text{ computer modules have successfully operated during } I_j \text{ up to the levels of degradation allowed, } 1 \le j \le J).$$

This term, $R(I_j, 1 \le j \le J)$ is called the <u>Mission Profile Reliability.</u> Here Mission (J) = $(I_1, \ldots I_J)$ is the mission whose J successive phases consist of $I_1 = I_1(T_0, T_1), \ldots I_J = I_J(T_{J-1}, T_J)$. Thus, for each design specification of tolerable mode degradation/module class/phase one may come up with a mission profile reliability. The analysis is thus set up to produce an evaluation tool which allows the system designer to vary the requirements of mission phase computer module use, the kind of redundancy he feels is desirable for that phase, and the type of mode degradation that he is prepared to live with on a phase by phase basis, thus allowing a large degree of design flexibility.

Throughout the following analysis we make the additional assumptions listed below:

1) Not only are computer module failure law distributions during the active and passive phases given by $F_1(t)$, $F_2(t)$ resp., but, in addition, if a module which has not failed is turned off during a given phase I_j and subsequently reconfigured into activity during I_k, $j<k$, then its failure law during the new dormant or active phase remains <u>invariant</u>. Thus, if $F_1(t) = 1 - e^{-\lambda t}$ was the active failure law, initially, for the given module, then at all subsequent resuscitations of that module (given no failure has previously occurred) the failure law will remain $F_1(t)$, with a similar rule pertaining to dormancy periods.
2) Inter and intra modular switching will be assumed to be perfect.
3) Fault detection and isolation for A(L, S) types of redundancy will be perfect. For TMR or NMR types of operation one may add a correction factor of $R_j(T_j - T_{j-1})$ = Prob (Voters operate successfully during mission phase I_j $[T_{j-1}, T_j]$).
4) All module failures occur independently of each other no matter what phase is being considered.

We next develop the Markov approach to MPR: Suppose the computer is partitioned into W module classes where the first W_1 module classes might be the same, the next W_2 are the same, etc. Each module class fails independently of all other module classes so that, if we label the module classes as $C_1, C_2, \ldots C_{W_1}, C_{W_1+1}, \ldots C_W$, it is irrelevant that the modules in either of the first W_1 classes are identical. The prescription, fixed in advance, of how many modules/class/partition are required in each given mission phase, allowing for mode degradation and different types of redundancy usage, enables the analysis to proceed on a per partitioned module subclass basis.

Thus, if $R(C_i)$ [T] = Reliability of module class partition C_i mission at some prescribed time T, then $R(I_1, \ldots I_J)$ = R(Mission (J)) and one may write:

$$R(C_1, C_2 \ldots C_W)[T] = \prod_{i=1}^{W} R(C_i)[T] = R(\text{Mission (J)}) \qquad \text{(Eq 6)}$$

It suffices then, to consider the problem for a specific module class (or module class partition) C which has N+S = M total modules of the same structure in it. One may assume that all modules are in the non-failed state at time t=0. Consider the Markov chain obtained by examining the module class C at the end of Phase $I_i = I_i[T_{i-1}, T_i]$. Let S_k^i = State that there are k non-failed modules among the M and that all mission phases from I_1 to I_i have been successfully performed at time T_i = instant after the mission phases I_j, for $1 \le j \le i$, are over.

Let $\pi_k(i)$ = Prob. that module class C is in state S_k^i at time T_i.

Suppose the mission phase requirement for phase I_{i+1} (T_i, T_{i+1}) is that N_{i+1} modules should be activitated during this period and if there are not that many non-failed modules available at time T_i, then all the available modules should be used but at no time can this number drop below D_{i+1}, the lowest (degraded) mode level allowable without having that mission phase being a failure. We seek the transition probability matrix for this phase of the Markov chain.

Clearly, we do not care whether we have TMR, NMR, etc. once N_{i+1}, D_{i+1} are specified; it is irrelevant from a reliability viewpoint what type of instrumentation or structuring is implemented. The system during I_{i+1} is in one of the hybrid types $H(N_{i+1}, \max(0, k-N_{i+1}), D_{i+1})$, if $k \geq N_{i+1}$, while if $k < N_{i+1}$, the analysis of Section IIA carries over with $\lambda_1 = k\lambda$, $\lambda_2 = (k-1)\lambda, \ldots, \lambda_k = \lambda$ substituted into Eqs 5a and 5b and N+S substituted by k in Eq 5g.

Thus, we have the hybrid types $H(N_{i+1}, k_{i+1}, D_{i+1})$ with $k_{i+1} = \max(k - N_{i+1}, 0)$, for $N_{i+1} \leq k$, and the hybrid types $H(k, 0, D_{i+1})$, for $D_{i+1} \leq k < N_{i+1}$. For $k < D_{i+1}$, there is clearly a module class failure.

Therefore, the Markov matrix equations may be written with associated transition probabilities as:

$$
\begin{bmatrix} \pi_0(i+1) \\ \vdots \\ \pi_{D_{i+1}-1}(i+1) \\ \pi_{D_{i+1}}(i+1) \\ \vdots \\ \pi_k(i+1) \\ \vdots \\ \pi_M(i+1) \end{bmatrix}
=
\begin{array}{c}
\begin{array}{ccccccccccc} \underline{0} & & \underline{D_{i+1}-1} & & \underline{D_{i+1}} & & \underline{k} & & \underline{i} & & \underline{M} \end{array} \\
\begin{bmatrix}
0 & \cdots & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 & & 0 \\
\vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
0 & \cdots & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 & & 0 \\
0 & \cdots & 0 & \cdots & M_{D_{i+1},D_{i+1}} & \cdots & m^{(i+1)}_{D_{i+1},k+1} & \cdot & m^{(i+1)}_{D_{i+1},j} & \cdots & m^{(i+1)}_{D_{i+1},M} \\
\vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
0 & \cdots & \cdot & \cdots & 0 & \cdots & m^{(i+1)}_{k,k} & \cdots & m^{(i+1)}_{k,j} & \cdots & m^{(i+1)}_{k,M} \\
\vdots & & & & & & \vdots & & \vdots & & \vdots \\
0 & \cdots & \cdot & \cdots & & & \cdot & \cdots & 0 & \cdots & m^{(i+1)}_{M,M}
\end{bmatrix}
\end{array}
=
\begin{bmatrix} \pi_0(i) \\ \vdots \\ \pi_{D_{i+1}-1}(i) \\ \pi_{D_{i+1}}(i) \\ \vdots \\ \pi_k(i) \\ \vdots \\ \pi_M(i) \end{bmatrix}
\quad \text{(Eq 7)}
$$

where,

$$\Delta T_i = T_i - T_{i-1}$$

and if $D_{i+1} = 0$,

$$m_{00}^{(i+1)} = 1 = P_0(0,0,0).$$

The general term is:

$$m_{kj}^{(i+1)} \begin{cases} = 0 \text{ for } k < D_{i+1},\ 0 \le j \le M \text{ or for } k \ge D_{i+1} \text{ but } j < D_{i+1} \\ = P_k\,(\min(j, N_{i+1}),\ \max(0,\ j-N_{i+1}),\ D_{i+1})\ [\Delta T_i] \\ \quad \text{for } k \ge D_{i+1},\ j \ge D_{i+1} \end{cases} \qquad \text{(Eq 8)}$$

$$\therefore\ \Pi(i+1) = \mathfrak{M}_{i+1}\Pi(i) \qquad \text{for} \qquad 0 \le i \le J-1 \qquad \text{(Eq 9a)}$$

where,

$$\Pi(i) = \begin{pmatrix} \pi_0(i) \\ \cdot \\ \cdot \\ \cdot \\ \pi_M(i) \end{pmatrix}$$ is the vector of i-th stage probabilities,

and $\mathfrak{M}_{i+1}$ = the matrix in the above system = $\left(m_{kj}^{(i+1)}\right)_{k,j=0}^{M}$.

$$\therefore\ \Pi(J) = (\mathfrak{M}_J \cdot \mathfrak{M}_{J-1} \cdot\ \cdot \mathfrak{M}_1)\ \Pi(0) = \mathfrak{M}^{(J)} \cdot \Pi(0) \qquad \text{(Eq 9b)}$$

where,

$$\Pi(0) = \begin{pmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 1 \end{pmatrix}$$

$$\therefore R(C)[T] = \sum_{i=0}^{M} \pi_i(J) \qquad \text{(Eq 9c)}$$

where,

$T = T_J$ is the endpoint of the mission.

At most, then, each module class (partition) will involve J matrix multiplications of order M+1 if the module class has M modules in it.

Since the matrices $\mathfrak{M}_i$ may have very many zeroes in them when D_i is large relative to M = N+S, where N and S are the initial numbers of working units and spares (respectively) either when the initial use was TMR, NMR or standby (active-passive) sparing, it is desirable to rework the form of Eq 7a.

It would be more efficient to write the matrix equations of Eq 7a in the form:

$$\pi_0(i+1) = \pi_1(i+1) = \ldots = \pi_{D_{i+1}-1}(i+1) = 0, \text{ while}$$

$$\Pi'(i+1) = \begin{bmatrix} \pi_{D_{i+1}}(i+1) \\ \cdot \\ \cdot \\ \cdot \\ \pi_M(i+1) \end{bmatrix} \qquad \text{(Eq 7a)}$$

is the reduced state space probability vector at level i+1 given by the matrix equation:

$$\Pi'(i+1) = \mathfrak{M}'_{i+1}\, \Pi'(i)$$

Hence, $\Pi'(i)$ is the reduced state space probability vector at level i, which is identical to $\Pi'(i)$ with the first $D_{i+1}-1$ terms removed, and $\mathfrak{M}'_{i+1}$ is the reduced $(M-D_{i+1}+1) \times (M-D_{i+1}+1)$ matrix whose terms are $m_{kj}^{(i+1)}$, where $D_i \le k$, $j \le M$ and $m_{kj}^{(i+1)}$ are the terms of the original matrix $\mathfrak{M}_{i+1}$.

B. The Space Station – A Special Case of Periodicity

As an example of a mission phase profile for which the above analysis may be presented in a simpler analytic form, we consider the case of a self-sustaining Space Station. Here, no repairs or logistic support is assumed possible. We wish to determine the reliability at the end of some specified time T. The development will be studied for a single module class, initially consisting of M modules.

The Space Station is visualized to operate in a periodic fashion. Thus, for some time $\alpha_1\phi$ the station will be in the high computational mode (HCM), and then for time $\alpha_2\phi$ in the high reliability mode (HRM), and finally for the time $\alpha_3\phi$ it will operate in the dormant mode (DM), in which everything is turned off. When in the HCM, everything is assumed to be on, while in HRM we assume that $D_w=3$ for module class w, $1 \le w \le W$. The α_i are positive constants satisfying the equation $\alpha_1+\alpha_2+\alpha_3=1$.

At the end of the DM operation the system will return to HCM, if possible, and the cycle will attempt to repeat itself with the same time constants. It is obvious that if we examine the case in which J is a multiple of three, i.e. $J = 3J_1$, then we have:

$$
\begin{aligned}
\Pi(J) &= \mathfrak{M}_J \cdot \mathfrak{M}_{J-1} \cdot \ldots \ldots \mathfrak{M}_3 \cdot \mathfrak{M}_2 \cdot \mathfrak{M}_1 \cdot \Pi(0) \\
&= \left(\mathfrak{M}_3 \cdot \mathfrak{M}_2 \cdot \mathfrak{M}_1 \right)^{J_1} \Pi(0)
\end{aligned}
$$

Therefore, letting $\mathfrak{M}^* = \left(\mathfrak{M}_3 \cdot \mathfrak{M}_2 \cdot \mathfrak{M}_1 \right)$, we have

$$\Pi(J) = (\mathfrak{M}^*)^{J_1} \Pi(0)$$

Since everything is turned on in HCM, we observe that the structure of the matrix $\mathfrak{M}_1$ is simply:

$$
\mathfrak{M}_1 = \begin{pmatrix} 0 & \cdots & & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \vdots & & 0 & 0 \\ \cdots & & 0 & m^{(1)}_{MM} \end{pmatrix}
$$

Therefore, the product matrix $\mathfrak{M}^*$ is of the form

$$
\mathfrak{M}^* = \left(\begin{array}{c|c} & x_1 \\ & x_2 \\ 0 & \vdots \\ & x_M \end{array} \right)
$$

, i.e. only the last column has potentially non-zero entries.

But then,

$$
(\mathfrak{M}^*)^2 = \left(\begin{array}{c|c} & x_1 x_M \\ & x_2 x_M \\ 0 & \vdots \\ & x_M x_M \end{array} \right)
$$

i.e. all columns, except possibly the last, are null columns.

By inspection, we find:

$$(\mathfrak{M}^*)^{J_1} = \left(\begin{array}{c|c} & x_1(x_M)^{J_1-1} \\ & x_2(x_M)^{J_1-1} \\ 0 & \vdots \\ & x_{M-1}(x_M)^{J_1-1} \\ & (x_M)^{J_1} \end{array} \right)$$

So that,

$$\Pi(J) = (\mathfrak{M}^*)^{J_1} \Pi(0) = \begin{pmatrix} x_1(x_M)^{J_1-1} \\ x_2(x_M)^{J_1-1} \\ \vdots \\ x_M(x_M)^{J_1-1} \end{pmatrix}$$

Therefore, if C is the underlying module class we obtain,

$$R(C)[T] = x_M^{J_1-1} \cdot \sum_{i=1}^{M} x_i$$

In the case of the self-sustaining Space Station, the effects of periodicity and the initial HCM mode combine to produce a total matrix sequence which is easily reduced for computational purposes.

IV. Future Effort

The method presented here represents an initial step in the reliability analysis of the ARMMS computer which will be used to compute predictions of reliability for sample mission sequences and module failure rates. Further refinements in the analysis will proceed in the direction of extending the model to include the following: 1) the effects of fault detection, fault location, and intramodular and intermodular switching; 2) the effects of voter failures and voter replication; 3) the effects of bussing failures and executive hardware failures; and 4) a treatment of non-exponential failure laws, in the case when the dormant and active failure law distributions are equal.

REFERENCES

(1) F.P. Mathur, "Reliability Modeling and Analysis of a Dynamic TMR System Utilizing Standby Spares," Proc Seventh Annual Allerton Conference on Circuit and System Theory, October 1969, P. 243-249.

(2) W.G. Bouricius, W.C. Carter and P.R. Schneider - "Reliability Modeling Techniques for Self-Repairing Computer Systems," ACM 1969 Annual Conference, P. 295-309. Also, in greater detail, as IBM Report Number RC-2378.

(3) J.K. Knox-Seith, "A Redundancy Technique for Improving the Reliability of Digital Systems," Stanford Electronics Laboratory, TR No. 4816-1, December, 1963.

(4) P.O. Nerber, "Power-Off Time Impact on Reliability Estimates," IEEE Int. Convention Record, Part 10, March 22-26, New York, P. 1-5.

(5) M. Ball and F.H. Hardie, "Architecture for an Extended Mission Aerospace Computer," IBM No. 66-825-1753, Owego, New York, May 1969.

(6) W.G. Bouricius, W.C. Carter, D.C. Jessep, P.R. Schneider, and A.B. Wadi, "Reliability Modeling for Fault-Tolerant Computers" - IEEE Transactions on Computers, Vol C-20, No. 11, November 1971.

(7) F.P. Mathur and A. Avizienis - "Reliability Analysis and Architecture of a Hybrid-Redundant Digital System: Generalized Triple Modular Redundancy with Self-Repair" - Proc 1970 Spring Joint Computer Conference, AFIPS Conf. Proc., Montvale, N.J.: AFIPS Press, May 1970, P. 375-383.

(8) F.P. Mathur - "On Reliability Modeling and Analysis of Ultrareliable Fault-Tolerant Digital Systems," IEEE Transactions on Computers, Nov, 1971 P. 1376-1382.

(9) E.J. Kletsky - "Upper Bounds on Mean Life of Self-Repairing Systems" - IEEE Trans. on Reliability and Quality Control, V. RQC-11, No. 3, Oct 1962.

(10) D.S. Taylor - "A Reliability and Comparative Analysis of Two Standby Configurations" - Performed Under NASA Contract NAS8-21805.

(11) J. L. Bricker – "Reliability Studies of the NASA Deep Space Computer and the H4400 Computer" – Appendix B of Proposal for a Study of the Increase of Aerospace Multiprocessor Life by Means of System Recovery from Failures Without Human Intervention – Hughes Aircraft Co. – GSG – 15 March 1969 ERC/R&D RCA-0030 B8964-100

CONSISTENCY OF R(N,S,D) WITH RESPECT TO STANDBY SPARING AND TMR/S

In this portion of the Appendix, we shall verify that the expressions for $R(N,S,D)\,[t]$, obtained from Eqs. 5a, 5b, 5d, and 5i, agree with those of TMR/S and standby sparing (active-standby passive redundancy) found in the literature (Eqs 1 and 3, respectively, of the text).

A. $\underline{A(N,S) \equiv H(N,S,N)}$

A. From Eq 3, taking $c = 1$, the case of perfect coverage, one finds that:

$$R_A(N,S)\,[t] = e^{-N\lambda t} \sum_{k=0}^{S} \binom{k-1+NK}{k} \cdot \left(1 - e^{-\mu t}\right)^k \qquad \text{(Eq 3)}$$

Now from Eq 5i, setting $D = N$, to make the $H(N,S,N)$ scheme correspond to $A(N,S)$, we see that

$$R(N,S,N)\,[t] = \sum_{j=1}^{N+S-D+1} A_j^{N+S-D+1}\, e^{-\lambda_j t}$$

$$= \sum_{j=1}^{S+1} A_j^{S+1}\, e^{-\lambda_j t} = e^{-N\lambda t} \cdot \sum_{j=1}^{S+1} A_j^{S+1}\, e^{-\lambda_j t} \qquad \text{(Eq 5i)}$$

$$A_j^{S+1} = \prod_{\substack{k \neq j \\ 1 \le k \le S+1}} \frac{\lambda_k}{(\lambda_k - \lambda_j)}$$

and $\lambda_k = N\lambda + (S - (k-1))\,\mu$ for $1 \le k \le S+1$

Expanding, Eq 3 yields,

$$R_A(N,S) = e^{-N\lambda t} \cdot \sum_{k=0}^{S} \binom{k-1+NK}{k} \sum_{r=0}^{k} (-1)^r \binom{k}{r} e^{-r\mu t}$$

$$= e^{-N\lambda t} \cdot \sum_{r=0}^{S} \sum_{k=r}^{S} e^{-r\mu t} (-1)^r \binom{k}{r} \binom{k-1+NK}{k} \qquad \text{(Eq 10)}$$

$$= e^{-N\lambda t} \cdot \sum_{r=0}^{S} e^{-r\mu t} (-1)^r \sum_{k=r}^{S} \binom{k}{r} \binom{k-1+NK}{k}$$

It suffices to show then, in Eq 5i, that

$$A_{S+1-r}^{S+1} = (-1)^r \cdot \sum_{k=r}^{S} \binom{k}{r} \cdot \binom{k-1+NK}{k} \qquad \text{(Eq 11)}$$

For r=0, this is equivalent to showing that

$$\sum_{k=0}^{S} \binom{k-1+NK}{k} = A_{S+1}^{S+1} = \prod_{\substack{k \neq S+1 \\ 1 \leq k \leq S+1}} \left[\frac{(\lambda_1) \cdot \cdots \cdot (\lambda_S)}{(\lambda_1 - \lambda_{S+1}) \cdots (\lambda_S - \lambda_{S+1})} \right] \qquad \text{(Eq 11a)}$$

$$= \frac{(N\lambda + S\mu) \cdots (N\lambda + \mu)}{(S\mu) \cdots (\mu)} = \frac{(1+KN) \cdots (S+KN)}{S!} = \binom{S+NK}{S}$$

$$B_r = \binom{r-1+NK}{r} (-1)^r \cdot \sum_{k=0}^{S-r} (-1)^k \binom{-NK-r}{k} \qquad \text{(Eq 12)}$$

$$= (-1)^r \binom{r-1+NK}{r} \sum_{k=0}^{S-r} \binom{k-1+NK+r}{k}$$

which, by (Eq 11b), is equivalent to:

$$B_r = (-1)^r \cdot \binom{r-1+NK}{r} \cdot \binom{NK+S}{S-r} \qquad \text{(Eq 13a)}$$

Now,

$$A^{S+1}_{S+1-r} = \frac{[(N\lambda + S\mu) \cdot \cdot \cdot \cdot \cdot (N\lambda + (r+1)\mu)]\ [(N\lambda + (r-1)\mu) \cdot \cdot \cdot \cdot \cdot (N\lambda)]}{[(S-r)\mu) \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot (\mu)] \quad \cdot \quad [(-\mu) \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot (-r\mu)]}$$

$$= \frac{[(S+NK) \cdot \cdot \cdot \cdot \cdot (r+1+NK)]}{\left[(S-r)\ !\right]} \cdot \frac{[(r-1)+NK) \cdot \cdot \cdot \cdot \cdot (NK)]}{\left[(-1)^r r!\right]}$$

$$= (-1)^r \binom{NK+r-1}{r} \cdot \binom{NK+S}{S-r} \qquad \text{(Eq 13b)}$$

$$B_r = A^{S+1}_{S+1-r}$$

and

$$R(N,S,N)[t] = R_A\,(N,S)[t] \qquad \text{(Eq 14)}$$

B. TMR/O ≡ H(3, 0, 2)

We shall first prove the equivalence of TMR/O = H(3, 0) with that of our scheme H(3, C, 2).

However, it is easy to show that,

$$\sum_{k=0}^{S} \binom{k-1+NK}{k} = \binom{S+NK}{S} \qquad \text{(Eq 11b)}$$

by induction on S.

Now consider the general case and note that, for k, r positive integers, with $k \geq r$, and α any real number, one always has the identity:

$$\binom{\alpha}{k} \cdot \binom{k}{r} = \binom{\alpha}{r} \cdot \binom{\alpha - r}{k-r},$$

and also,

$$\binom{k-1+NK}{k} = (-1)^k \cdot \binom{-NK}{k} .$$

It then follows that:

$$B_r = (-1)^r \cdot \sum_{k=r}^{S} \binom{k}{r} \cdot \binom{k-1+NK}{k}$$

$$= (-1)^r \cdot \sum_{k=r}^{S} \binom{k}{r} \cdot \binom{-NK}{k} (-1)^k$$

$$= (-1)^r \cdot \binom{-NK}{r} \cdot \sum_{k=r}^{S} (-1)^k \binom{-NK-r}{k-r}$$

Now,

$$R_{TMR/O}(t) = 3e^{-2\lambda t} - 2e^{-3\lambda t} = 3R^2 - 2R^3 \qquad \text{(Eq 15)}$$

where,

$$R = e^{-\lambda t}$$

as is obvious from inspection.

Let us check R(3, 0, 2)[t] to verify that we obtain the same result.

From Eq 5i we have:

$$R(3,0,2)[t] = \sum_{j=1}^{2} A_j^2 \, e^{-\lambda_j t}$$

where

$$\lambda_1 = 3\lambda,$$

$$\lambda_2 = 2\lambda,$$

while

$$A_1^2 = \frac{\lambda_2}{\lambda_2 - \lambda_1} = -2$$

$$A_2^2 = \frac{\lambda_1}{\lambda_1 - \lambda_2} = 3$$

Therefore $R(3,0,2)[t] = 3e^{-2\lambda t} - 2e^{-3\lambda t} = R_{TMR/O}(t)$ (Eq 16)

C. $\underline{H(3,S) = H(3,S,2)}$

We now treat the general case of TMR/S.

From (Eq 1) of the text we find:

$$R(3,S)[t] = 3e^{-2\lambda t} \left\{ \left(\prod_{i=0}^{S-1} \frac{3K+S-i}{K+S-i} \right) - e^{-\lambda t} \frac{2K^2}{S\,!} \left[\prod_{i=0}^{S-1} (3K+S-i) \right] \right.$$

$$\left. \cdot \sum_{i=0}^{S} \binom{S}{i} \frac{(-e^{-\mu t})^{S-i}}{(K+S-i)(3K+S-i)} \right\} \qquad \text{(Eq 1)}$$

Comparing this with (Eq 5i) it is then sufficient to show that:

$$R(3,S)[t] = R(3,S,2)[t] = \sum_{i=1}^{S+2} A_j^{S+2} e^{-\lambda_j t}$$

where,

$$A_j^{S+2} = \prod_{\substack{k \neq j \\ 1 \leq k \leq S+2}} \frac{\lambda_k}{(\lambda_k - \lambda_j)}$$

and,

$$\lambda_k = 3\lambda + (S-k+1)\mu, \qquad \text{for } 1 \leq k \leq S+1$$

$$\lambda_{S+2} = 2\lambda$$

By matching up coefficients of the exponential terms, one sees that one must verify that,

a) $$3 \cdot \prod_{i=0}^{S-1} \frac{3K + S-i}{k + S-i} = A_{S+2}^{S+2},$$

b) $$B = \left[\frac{-6K^2}{S} \prod_{i=0}^{S-1} (3K + S-i)\right]\binom{S}{j-1}\frac{(-1)^{S-j+1}}{(K+S - j+1)\ (3K+S - j+1)} = A_j^{S+2}$$

for $1 \le j \le S+1$.

Now,

$$A_{S+2}^{S+2} = \frac{(3\lambda + S\mu) \cdots (3\lambda + \mu)\ (3\lambda)}{(\lambda + S\mu) \cdots (\lambda + \mu)\ (\lambda)} = 3 \prod_{i=0}^{S-1} \frac{3K + S-i}{K + S-i}$$

establishing a).

For b), we find that for $1 \le j \le S+1$,

$$A_j^{S+2} = \prod_{\substack{k \ne j \\ 1 \le k \le S+2}} \left(\frac{\lambda_k}{\lambda_k - \lambda_j}\right)$$

$$= \frac{(3\lambda + S\mu) \cdots (3\lambda + (S - j+2)\mu) \cdot (3\lambda + (S-j)\mu) \cdots (3\lambda + \mu)\ (3\lambda)\ (2\lambda)}{(j-1)\mu) \cdots (\mu) \cdots (-\mu) \cdots (-(S + 1-j)\mu) \cdot (-\lambda - (S - j+1)\mu)}$$

Multiplying numerator and denominator by the missing term $3\lambda + (S - j+1)\mu$ yields,

$$A_j^{S+2} = (-1)^{S-j}\, 6K^2 \frac{\prod_{i=0}^{S-1} (3K + S-i)}{(j-1)\,!\,(S+1-j)\,!\,(K+S-j+1)(3K+S-j+1)}$$

$$= -\frac{6K^2}{S!}\binom{S}{j-1} \cdot \prod_{i=0}^{S-1} (3K + S-i) \left((-1)^{S-j+1} \frac{1}{(K+S-j+1)\,(3K+S-j+1)}\right)$$

which is exactly the term B in b) above.

Therefore, $R(3,S)[t], \equiv R(3,S,2)[t]$, which agrees with Mathur's result as stated in [1].

The proof that the NMR/S or H(N,S) scheme is equivalent to H(N,S, n+1) for N=2n+1 is considerably more involved, and was omitted to save space; it was also verified by a computer program. It is quite clear by looking at Eq 2 of the text and comparing it with Eq 5i, that the results obtained here are easier to grasp and manipulate mathematically. The fundamental idea is that one is dealing with sums of independent, but not necessarily identically distributed, negative exponential random variables, and this is the unifying principle which unites the previous, apparently disconnected, results of [1], [2], [6], [7], [8], [9] and [10] into one simple result.